日 本 国 特 許 庁
PATENT OFFICE
JAPANESE GOVERNMENT

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日
Date of Application: １９９９年１１月１７日

出願番号
Application Number: 平成１１年特許願第３２７２７６号

出願人
Applicant (s): インターナショナル・ビジネス・マシーンズ・コーポレイション

２０００年 ３月１０日

特許庁長官
Commissioner,
Patent Office 近藤隆彦

出証番号 出証特２０００－３０１４４６５

【書類名】　特許願

【整理番号】　JA999195

【あて先】　特許庁長官　殿

【国際特許分類】　G06F　9/44

【発明者】

【住所又は居所】　神奈川県大和市下鶴間１６２３番地１４　日本アイ・ビー・エム株式会社　東京基礎研究所内

【氏名】　田井　秀樹

【発明者】

【住所又は居所】　神奈川県大和市下鶴間１６２３番地１４　日本アイ・ビー・エム株式会社　東京基礎研究所内

【氏名】　山本　学

【発明者】

【住所又は居所】　神奈川県大和市下鶴間１６２３番地１４　日本アイ・ビー・エム株式会社　東京基礎研究所内

【氏名】　中村　祐一

【特許出願人】

【識別番号】　390009531

【住所又は居所】　アメリカ合衆国１０５０４、ニューヨーク州アーモンク（番地なし）

【氏名又は名称】　インターナショナル・ビジネス・マシーンズ・コーポレイション

【代理人】

【識別番号】　100086243

【弁理士】

【氏名又は名称】　坂口　博

【連絡先】　０４６２－７３－３３１８、３３２５、３４３１

【選任した代理人】

【識別番号】　100091568

【弁理士】

【氏名又は名称】　市位　嘉宏

【手数料の表示】

【予納台帳番号】　024154

【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【包括委任状番号】　9304391

【包括委任状番号】　9304392

【プルーフの要否】　要

【書類名】　明細書

【発明の名称】　移動エージェント管理装置およびその方法

【特許請求の範囲】

【請求項１】

複数のエージェントサーバと、移動エージェントの位置を保持する登録サーバとを含む移動エージェント管理装置であって、

前記エージェントサーバそれぞれは、

移動エージェントの移動履歴を保持する移動履歴保持手段と、

移動エージェントそれぞれの位置情報の更新要求を保持する更新保持手段とを有し、

保持された前記更新要求を、前記登録サーバが保持する移動エージェントの位置情報に定期的に反映させる

移動エージェント管理装置。

【請求項２】

前記エージェントサーバそれぞれは、前記更新要求を、対応する移動エージェントの移動回数と対応づけて前記登録サーバに送信し、

前記登録サーバは、移動エージェントそれぞれの移動回数と位置とを対応づけて保持し、より多い移動回数と対応づけられた前記更新要求のみを、移動エージェントそれぞれの位置情報に反映させる

請求項１に記載の移動エージェント管理装置。

【請求項３】

複数のエージェントサーバと、移動エージェントの位置を保持する登録サーバとを用いて移動エージェントの位置を管理する移動エージェント管理方法であって、

前記エージェントサーバそれぞれにおいて、

移動エージェントの移動履歴を保持し、

レジストリの更新要求および削除要求を保持し、

保持された前記更新要求および削除要求を、定期的に前記登録サーバが保持する移動エージェントの位置情報に反映させる

移動エージェント管理方法。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】

本発明は移動エージェントの位置を管理する移動エージェント管理装置およびその方法に関する。

【０００２】

【従来の技術】

移動エージェントの管理を行う場合や、ユーザが過去に放った自分のエージェントへメッセージを送信したり引き戻す場合には、移動エージェントが今どこにいるのかという情報を管理する必要がある。

例えば、「GMD FOKUS and IBM Corp., Mobile Agent System Interoperability FacilitiesSpecification, OMG TC Document(orbos/97-10-05), Nov 1997（[MASIF]；文献１）は、ロギング(Logging)により移動エージェントの位置を追跡する方法を開示する。

【０００３】

また、上記文献１および「GrassHopper - A Mobile Agent Platform for IN Based Service Environments, Proceedings of IEEE IN Workshop 1998, pp. 279-290, May 1998（[GrassHopper]；文献２）は、レジストリ(Registry)により移動エージェントの位置を追跡する方法を開示する。

【０００４】

一方、移動エージェントの位置管理には、その性質上、（１）問い合わせに対する情報の即時性。今現在の情報が提供できること、（２）追跡可能性の高さ。分散環境での障害に際し、できるだけ多くのエージェントが追跡可能であること。また復旧までの時間ができるだけ短いこと。（３）スケーラビリティ。より大きな数のエージェントや、頻繁な移動に対応できること。（４）情報管理のためのコストが低いこと。エージェントの移動に伴うオーバーヘッドが小さいこと等が求められる。

しかしながら、上述した文献１，２に開示された方法は、これらの要求を充分

に満たしてはいない。

【０００５】

【発明が解決しようとする課題】

本発明は、上述した従来技術の問題点に鑑みてなされたものであり、より高い次元で移動エージェントの位置管理に求められる要求を満たす移動エージェント管理装置およびその方法を提供することを目的とする。

【０００６】

【課題を達成するための手段】

上記目的を達成するために、本発明に係る移動エージェント管理装置は、複数のエージェントサーバと、移動エージェントの位置を保持する登録サーバとを含む移動エージェント管理装置であって、前記エージェントサーバそれぞれは、移動エージェントの移動履歴を保持する移動履歴保持手段と、移動エージェントそれぞれの位置情報の更新要求を保持する更新保持手段とを有し、保持された前記更新要求を、前記登録サーバが保持する移動エージェントの位置情報に定期的に反映させる。

【０００７】

好適には、前記エージェントサーバそれぞれは、前記更新要求を、対応する移動エージェントの移動回数と対応づけて前記登録サーバに送信し、前記登録サーバは、移動エージェントそれぞれの移動回数と位置とを対応づけて保持し、より多い移動回数と対応づけられた前記更新要求のみを、移動エージェントそれぞれの位置情報に反映させる。

【０００８】

また、本発明に係る移動エージェント管理方法は、複数のエージェントサーバと、移動エージェントの位置を保持する登録サーバとを用いて移動エージェントの位置を管理する移動エージェント管理方法であって、前記エージェントサーバそれぞれにおいて、移動エージェントの移動履歴を保持し、レジストリの更新要求および削除要求を保持し、保持された前記更新要求および削除要求を、定期的に前記登録サーバが保持する移動エージェントの位置情報に反映させる。

【０００９】

【発明の実施の形態】

本発明にかかる移動エージェント管理方法は、エージェントの位置を集中管理する登録サーバ（レジストリ）と、各エージェントサーバが残す移動履歴（ログ）とを組み合わせて、これら２つの方法がそれぞれ有する問題点を解決し、移動エージェントの位置管理に要求される用件を、より高い次元で満たすように構成されている。

本発明に係る移動エージェント管理方法の説明に先立ち、まず、移動エージェントの一般的な移動の仕組み、および、レジストリとロギングについて説明する。

【００１０】

図１は、移動エージェントとエージェントサーバの関係を表すものである。

一般的に移動エージェント(1)は、分散環境内でユニークなID(2)を持ち、その実行環境としてエージェントサーバ(3)といったランタイム(サーバープロセス)を必要とする。

各エージェントサーバは、URLなどを用いたアドレス(4)で特定される。エージェントサーバは移動エージェントを管理し、その実行や移動を行う。移動エージェントにとって、エージェントサーバが移動先の対象となる。

【００１１】

移動エージェントを移動させるためには、移動元のエージェントサーバは次のような処理を行う(図２)。

移動エージェントの実行を停止し、リストからその移動エージェントのエントリを削除する(1)。

移動エージェントの状態をバイト列に変換する(2)

移動エージェントのプログラムと状態を変換したバイト列を移動先のエージェントサーバに転送する(3)。

【００１２】

また、移動先のエージェントサーバは、移動元からの要求を受け、次のような処理で移動エージェントを生成する。

移動元のエージェントサーバから、移動エージェントのプログラムとバイト列

を受信する(4)。

受信したバイト列から、移動エージェントの状態を復元する(5)。

リストに復元した移動エージェントのエントリを追加し、その移動エージェントの実行を再開する(6)。

これらの移動元と移動先のエージェントサーバの処理により、移動エージェントの移動が完了する。

【００１３】

[移動エージェントの追跡（レジストリ）]

次に、移動エージェントを追跡する方法の一つ、レジストリを説明する（図３）。

これは、エージェントが移動するたびに登録サーバに新しい位置を記録するものである。 登録サーバは(7)、移動エージェントの移動範囲内に一つ存在し、移動エージェントのID(AgentID)とその位置(Address)のからなるテーブル(8)を管理している。

【００１４】

移動エージェントの位置を知るには、登録サーバに問い合わせればよい（図４）。

移動エージェントの位置を知りたいクライアント(9)は、その移動エージェントのIDをパラメータとして登録サーバに問い合わせる(10)。

問い合わせを受けた登録サーバ(11)は、その管理テーブル(12)から該当するエントリを検索し、その結果(13)を返答する。管理テーブルにエントリが見つからない場合は、「移動エージェントは見つからない」旨を返答する。

【００１５】

登録サーバの管理テーブルの更新は、移動エージェントが生成された時、消滅した時、移動した時に行われる（図５～７）。そのタイミングの詳細は実装によって異なるが、主に次のようなものになる。

移動エージェントの生成時(14)：その実行をエージェントサーバが開始する直前に、登録サーバに更新要求(15)を送り、登録サーバが管理テーブルを更新する(16)。

【００１６】

移動エージェントの移動時(17)：エージェントが移動先のエージェントサーバに到着(18)し、その実行を再開する直前に、登録サーバに新たな移動エージェントの位置の登録要求を送信(19)する。要求を受けた登録サーバは、該当する管理テーブルのエントリを書き換える(20)。

移動エージェントの消滅時(21)：その実行を停止する直前に、登録サーバに抹消要求(22)を送信し、登録サーバが管理テーブルから該当するエントリを削除する(23)。

【００１７】

レジストリはシンプルで実装も容易であるが、次のような問題がある。

（１）エージェントの移動のたびに登録サーバへ新しい位置を登録するオーバーヘッドが生じる。

（２）登録サーバへのアクセス集中によるパフォーマンスが低下する。

【００１８】

[移動エージェントの追跡（ロギング）]

次に、その他の移動エージェント追跡方法として、ロギングを説明する（図８，９）。

これは、エージェントが移動するたび(24,25)に、移動元のエージェントサーバに移動履歴を残しておき(26,27)、問い合わせ(29)の時には移動履歴をたどる(30)というものである。 移動履歴(28)は、以下のようなエントリから成るテーブルである。

【００１９】

【表１】

[AgentID, DestinationAddress]　　（１）

但し、AgentIDは移動エージェントのIDであって、このテーブルのキーとして用いられ、DestinationAddressはそのエージェントの移動先のエージェントサーバを特定するアドレスである。

【００２０】

問い合わせ、すなわち追跡要求は、移動履歴に従いエージェントサーバ間で次

々に渡され、最終的に移動エージェントが存在するエージェントサーバにたどり着いたところ(31)で、そのエージェントサーバのアドレスが追跡結果として返答される(32)。

【００２１】

以下では、エージェントサーバが行う移動履歴の管理について説明する。

エージェントサーバが移動履歴の更新を行うのは、移動エージェントが他のエージェントサーバに移動した場合（図１０）と、消滅した移動エージェントに関するエントリの削除要求を受け取った場合（図１１）である。

【００２２】

移動エージェントが移動を開始する時(33)には、移動先のエージェントサーバのアドレスを記したエントリ(34)を移動元の移動履歴に残す(35)。移動エージェントが消滅した時(36)には、移動元のエージェントサーバ(37)に対して削除要求(38)を送る。削除要求を受けたエージェントサーバ(37)は、自分の移動履歴から該当するエントリを削除し(39)、さらに移動元のエージェントサーバに同じ削除要求を送信する(40)。以下、移動履歴の連鎖の開始点(41)まで削除要求の送信を繰り返す。

【００２３】

ロギングは、移動エージェントの移動時の処理がエージェントサーバでのローカルな軽い処理だけで済むため、移動のオーバーヘッドが少ない。また、Registryのようなサーバへのアクセス集中が発生しないという利点がある一方、次のような問題がある。

（１）移動エージェントの移動履歴の連鎖上のエージェントサーバのうち、一つでも障害が発生すると、そのエージェントは追跡不可能になる。

（２）移動エージェントの移動履歴の連鎖上のエージェントサーバからでないと、その移動エージェントが追跡できない。

（３）移動エージェントが消滅した時に、移動履歴を消去する必要があり、頻繁にエージェントが消滅した場合にそのコストが大きい。

【００２４】

以下、本発明に係る移動エージェント管理方法を説明する。

本発明は、RegistryとLoggingを組み合わせることでスケーラビリティの向上と、常に移動エージェントの最新の位置の問い合わせを可能にするものである。本発明で想定するシステム環境（図１２）は、上述したレジストリと同様に、複数のエージェントサーバ(42,43,44)と、一つの登録サーバ(45)から成る。

【００２５】

各エージェントサーバ(42,43,44)は、上述したロギングと同様に、移動履歴(46)を保持し管理する機能を持つ。

ただし本発明では、図１３に示すように、各エージェントサーバは要求送信バッファ(47)も持つ。要求送信バッファは、Registryでの更新要求(15,19)や削除要求(22)に相当する情報をまとめて、定期的に送るために一時的に貯えておく場所である。

【００２６】

[要求のバッファリング]

レジストリでは、登録サーバが管理する分散環境の中で移動エージェントに起こるイベントの頻度が多くなると、登録サーバへリクエストが集中しパフォーマンスに問題が生じてしまい、スケーラブルであるとは言えない。

これは、個々の移動エージェントに生成、移動、消滅といったイベントが起こる毎に登録サーバへの情報更新が発生するためである。この問題を解決するために、本方法では個々の移動エージェント毎に位置情報を更新する事を避け、一度エージェントサーバに位置情報の更新要求(48,49)や抹消要求(50)を、要求送信バッファ(51)にバッファリングする。

【００２７】

要求送信バッファの内容は、定期的に登録サーバの管理テーブルに反映させられる(52)。要求のバッファリングにより、登録サーバへのリクエスト頻度を一定に制御することが可能になり、スケーラビリティを向上させることができる。

【００２８】

[ロギングとの組み合わせ]

本方法では要求をバッファリングために登録サーバの情報の更新に遅延ができてしまい、登録サーバの情報が常に最新のものではなくなってしまうという問題

が生じる。

この問題に対処するためには、Loggingと同様に移動履歴をエージェントサーバに残すようにする。その結果、登録サーバの位置情報を起点として移動履歴をたどることで、常に最新の位置情報を問い合わせることが可能になる。

【００２９】

要求のバッファリングにより生じるもう一つの問題は、登録サーバへ要求が送られる順番がまちまちになってしまう可能性である（図１４）。

これは、より新しい位置情報がより古い位置情報で上書きされてしまう問題に対処しなければならないことを意味する。例えば、移動エージェントがエージェントサーバ2で生成され(53)、エージェントサーバ1へ移動した(54)場合を考える。この時各エージェントサーバの要求バッファには登録要求がバッファリングされている(55,56)が、もしエージェントサーバ1の要求バッファの内容(55)が先に登録サーバに反映され、エージェントサーバ2の要求バッファの内容(56)が後に反映された場合、より新しいエージェントサーバ1の情報(57)が、より古いエージェントサーバ2の情報(58)で上書きされてしまう。

【００３０】

より古い情報での上書きが生じると、登録サーバの情報を基点とする移動履歴の連鎖が長くなり、追跡のコストが増してしまうだけでなく、エージェントサーバが移動履歴を保持しておかなければならない期間が長くなってしまい、より多くの移動履歴を保持しておかなければならなくなる。

【００３１】

本方法では、このような問題に対処するために、要求の世代という概念を導入する（図１５）。

要求の世代には、その要求が作られた時点での移動エージェントの移動回数を用いることができる。この場合、より大きな値がより新しい世代を表す。要求に世代の情報を含めることで(59,60)、登録サーバがより古い情報(61)で既に管理テーブルに登録されているより新しい情報(62)を上書きしてしまうことを防ぐことができる。

【００３２】

［実施例］

以下、本発明に係る移動エージェント管理方法の実施例として、各エージェントサーバにバッファリングする位置情報と登録サーバの管理情報、移動履歴について述べた後、これらの情報の更新アルゴリズム、及び移動エージェントを追跡するアルゴリズムを説明する。

登録サーバとエージェントサーバが管理する情報

本方法では、移動エージェントを生成した時や移動の結果到着した時、消滅した時に、次のような情報を登録要求としてエージェントサーバの要求バッファに作成する（図１６）。

【００３３】

【表２】

[AgentID, (Kind, HopCount)]　　（２）

【００３４】

但し、AgentIDは該当する移動エージェントのIDで、HopCountはこの情報を作成した時点の移動エージェントの、のべ移動回数を表す。

例えば、生成時にはHopCountには0が入る。Kindには、REGISTかUNREGISTといった値を取り、移動エージェントを生成した時と到着した時にREGIST、消滅した時にはUNREGISTとする。

【００３５】

登録サーバは次のようなエントリから成るテーブルを管理する（図１７）。

【００３６】

【表３】

[AgentID, (Address, HopCount, InUse)]　　（３）

【００３７】

但し、AgentIDは該当する移動エージェントのIDで、Addressはエージェントサーバのアドレスで、その移動エージェントの検索開始点を示す。Addressが空の場合、そのエントリが削除の候補となっている事を示す。また、HopCountは、この情報の世代を表すもので、値が大きいほど新しい情報であるものである事を示す。

具体的には、情報が作成された時点の移動エージェントの移動回数を用いる。InUseには整数値が入り、1以上の時にはこのエントリが移動エージェント追跡のために使用中である事を示す。

【００３８】

エージェントサーバは、登録要求の他、移動履歴も管理する。移動履歴は移動エージェントがそのエージェントサーバから移動した時に生成され、次のような情報を含む（図１８）。

【００３９】

【表４】

[AgentID, Destination]　　（４）

【００４０】

AgentIDは該当する移動エージェントのID，Destinationは移動先のエージェントサーバのアドレスである。

【００４１】

[管理情報の更新アルゴリズム]

はじめに、各エージェントサーバはその起動時に分散環境内の登録サーバに参加の意志を伝える（図１９）。

これを受けた登録サーバは、参加している各エージェントサーバのリストを管理し、各エージェントサーバに定期的に要求バッファの内容の送信をリクエストする（図２０）。

【００４２】

ここではこのエージェントサーバに対するリクエストを"sendBufferInfo"と呼ぶことにする。sendBufferInfoをリクエストする時には、管理テーブルのエントリのうち、InUseの値が1以上で、しかもAddress部がリクエストを送信しようとしているエージェントサーバのアドレスであるものを集め、sendBufferInfoリクエストのパラメータとする。sendBufferInfoの返答は登録サーバの管理テーブルに反映されるが、この時HopCountがより大きい場合にのみエントリの更新を行うようにする。

【００４３】

これは常により新しい情報を管理テーブルに反映させるためである。ただしKindがUNREGISTである場合には、これが最新かつ最後の情報であり、より新しい情報はありえないので、HopCountの値に関わらず即座に管理テーブルに反映させる。また、更新するエントリのAddress部には、KindがREGISTである要求に関しては、返答を返したエージェントサーバのアドレスをセットし、KindがUNREGISTの場合には空の値をセットする。

【００４４】

最後に、参加している全てのエージェントサーバにsendBufferInfoリクエストの送信とその返答の処理を行った後、登録サーバは不要なエントリを削除する。不要なエントリとは、削除の候補となっているもの、すなわちAddress部が空のエントリである。登録サーバが行うこの一連の処理のアルゴリズムを図２１～２３に示す。

【００４５】

sendBufferInfoのリクエストを受けたエージェントサーバは、図２４に示すアルゴリズムに従い、不要な移動履歴のエントリを消去した後、要求バッファの内容を登録サーバに返答する。またこの時、要求バッファを空にする。

【００４６】

[移動エージェント追跡アルゴリズム(図２５～２８)]

移動エージェントの追跡は、まず登録サーバに問い合わせ"lookupAgent"を発行し、追跡の開始点を得る。登録サーバは、lookupAgentを受けた時に管理テーブルの該当するエントリのInUseフィールドの値を1増やす。これは、追跡を行っている間に、sendBufferInfoリクエストを受けたエージェントサーバが移動履歴を破棄してしまうのを避けるためである。

InUseの値に関しては、移動エージェントの追跡が完了した後、登録サーバにその旨を伝えることで1減らすというのが最も単純な方法である。この他にも、一定時間たつと自然に値が1減ってしまうようにしておく方法もある。この場合、問い合わせ"lookupAgent"の発行者は、追跡が完了するまでは定期的にInUseフィールドの値を1増やすためのリクエストを登録サーバに送るようにしなければならない。

【００４７】

登録サーバの"lookupAgent"に対する返答が得られたならば、移動エージェントの追跡を開始する。つまり、追跡開始点のエージェントサーバに対して追跡要求を出す。しかし、場合によっては"エントリは見つからない"といった返答もありうる。この場合、移動エージェントは元々存在しないか、すでに消滅していることが考えられる。追跡要求の結果が、この移動エージェント追跡の結果となる。

追跡要求を受けたエージェントサーバは、自サーバ内に該当する移動エージェントを探し、見つかった場合には自サーバのアドレスを返答する。見つからない場合には、移動履歴から該当するエントリを探し、そこで示される移動先のエージェントサーバに追跡要求を出し、その結果を自分への追跡要求の返答とする。もし移動履歴にエントリが見つからない場合は、何らかの障害により移動履歴の連鎖が切れてしまっていることが考えられる。

【００４８】

[効果]

移動エージェントの位置問い合わせに答える機構としては、主に登録サーバベースのもの（Registry）が考えられている[MASIF, GrassHopper]。登録サーバは一種のネームサーバで、エージェントが移動するたびに新しい位置を登録するものである。しかしながら、Registryには、エージェントの移動のたびに登録サーバへ新しい位置を登録するオーバーヘッド登録サーバへのアクセス集中によるパフォーマンスの低下という問題がある。

【００４９】

このためRegistryでは、一つの登録サーバで管理する範囲を小さくするなどして、管理するエージェントの数を減らし、登録サーバを分散化するといった工夫が必要になる。また、登録サーバ同士を連携させて、管理の範囲を広げるということも必要になるが、その機構の具体的な仕組みは提案されていない。本発明で提案する方法は、Regstryを拡張したものであるが、Registryに対して次のような利点を持つ。

（１）情報の登録にかかるオーバーヘッドが少ない。

（２）Registryよりも大量のエージェントの位置管理が可能。

（３）メッセージの配送にも利用可能。

【００５０】

［評価］

本発明をAglets上に実装し、その性能評価を行った。

この実験では、エージェントサーバの数をNとし、最初各エージェントサーバ毎に10のエージェントを配置した。すなわち、系全体では10Nのエージェントが存在する。各エージェントは1秒おきにランダムに決定したエージェントサーバへの移動を繰り返し、全てのエージェントが20回の移動を終えるまでの時間(単位ms)を計測した。

【００５１】

なお、本発明の機構に特有なパラメータには、次のような値を設定した。

【００５２】

【数１】

$T_{expire}=6000, T_{update}=3000, P=1$

【００５３】

計測は従来の方法(Registry)，本方法，および全く追跡機構を伴わないものの3つの方法に関して各3回ずつ行った。

計測結果Case 1: N=9(エージェント総数90)、Case 2: N=15(エージェント総数150)を、それぞれ図２９，３０に示す。

【００５４】

【発明の効果】

以上述べたように、本発明に係る移動エージェント管理装置およびその方法によれば、上述した従来技術の問題点に鑑みてなされたものであり、より高い次元で移動エージェントの位置管理に求められる要求を満たすことができる。

【図面の簡単な説明】

【図１】

移動エージェントとエージェントサーバの関係を表す図である。

【図２】

移動エージェントを移動させるために、移動元のエージェントサーバが行う処理を示す図である。

【図３】

移動エージェントを追跡する第１の方法（レジストリ）を示す図である。

【図４】

移動エージェントの位置を知るための登録サーバへの問い合わせを示す図である。

【図５】

登録サーバの管理テーブルの更新を示す第１の図である。

【図６】

登録サーバの管理テーブルの更新を示す第２の図である。

【図７】

登録サーバの管理テーブルの更新を示す第３の図である。

【図８】

移動エージェントを追跡する第２の方法（ロギング）を示す第１の図である。

【図９】

移動エージェントを追跡する第２の方法（ロギング）を示す第２の図である。

【図１０】

移動エージェントが他のエージェントサーバに移動した場合のエージェントサーバが移動履歴の更新を示す図である。

【図１１】

消滅した移動エージェントに関するエントリの削除要求を受け取った場合のエージェントサーバが移動履歴の更新を示す図である。

【図１２】

本発明で想定するシステム環境を示す図である。

【図１３】

各エージェントサーバが有する要求送信バッファを示す図である。

【図１４】

登録サーバへ要求が送られる順番がまちまちになってしまう場合を示す図であ

る。

【図１５】

要求の世代という概念を示す図である。

【図１６】

登録要求としてエージェントサーバの要求バッファに作成される情報を示す図である。

【図１７】

登録サーバが管理するテーブルに含まれるエントリを示す図である。

【図１８】

移動履歴に含まれる情報を示す図である。

【図１９】

各エージェントサーバが起動時に分散環境内の登録サーバに参加の意志を伝える処理を示す図である。

【図２０】

各エージェントサーバに定期的に要求バッファの内容の送信をリクエストする処理を示す図である。

【図２１】

登録サーバが行うこの一連の処理のアルゴリズムを示す第１の図である。

【図２２】

登録サーバが行うこの一連の処理のアルゴリズムを示す第２の図である。

【図２３】

登録サーバが行うこの一連の処理のアルゴリズムを示す第３の図である。

【図２４】

sendBufferInfoのリクエストを受けたエージェントサーバの処理を示す図である。

【図２５】

移動エージェント追跡アルゴリズムを示す第１の図である。

【図２６】

移動エージェント追跡アルゴリズムを示す第２の図である。

【図２７】

移動エージェント追跡アルゴリズムを示す第３の図である。

【図２８】

移動エージェント追跡アルゴリズムを示す第４の図である。

【図２９】

本発明に係る移動エージェント管理方法の評価結果を示す第１の図である。

【図３０】

本発明に係る移動エージェント管理方法の評価結果を示す第２の図である。

【符号の説明】

４２，４３，４４・・・エージェントサーバ

４５・・・登録サーバ

４６・・・移動履歴

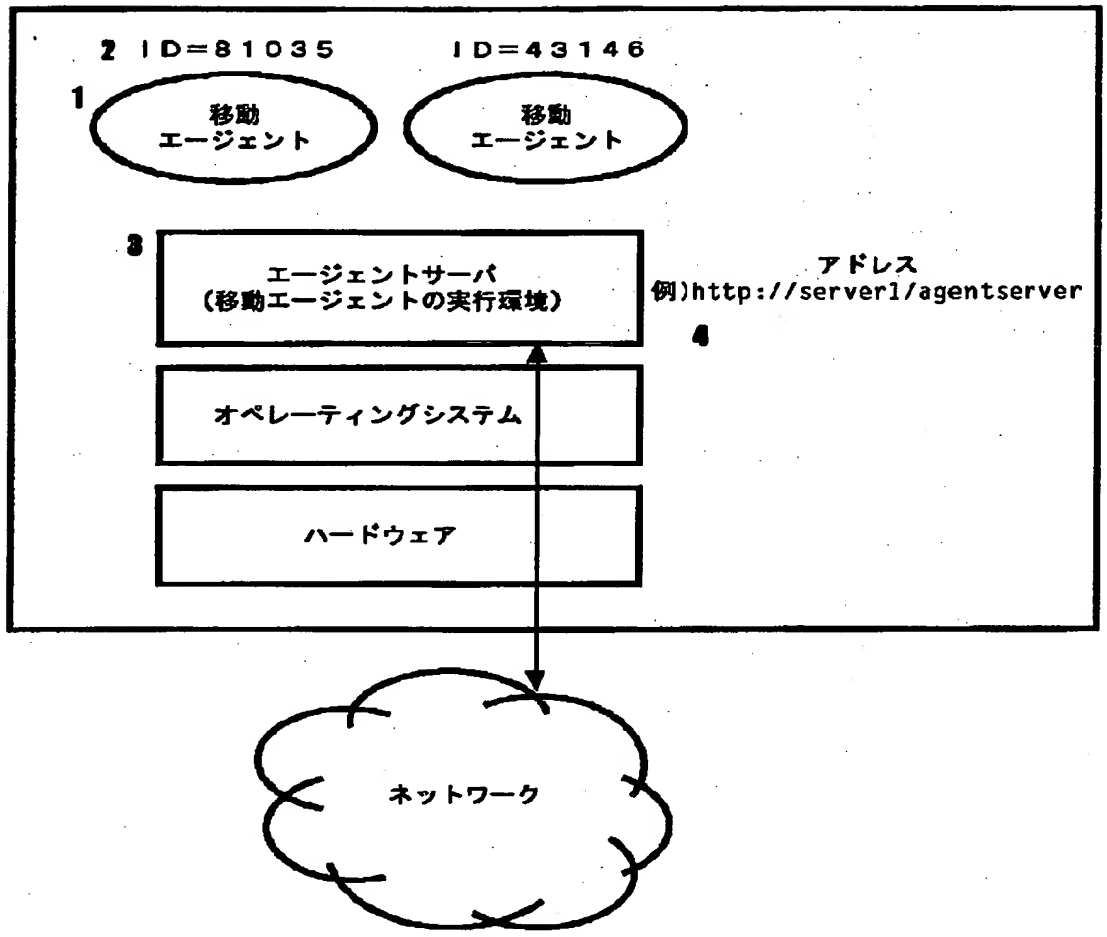
４７・・・要求送信バッファ

１５，１９・・・更新要求

２２・・・削除要求

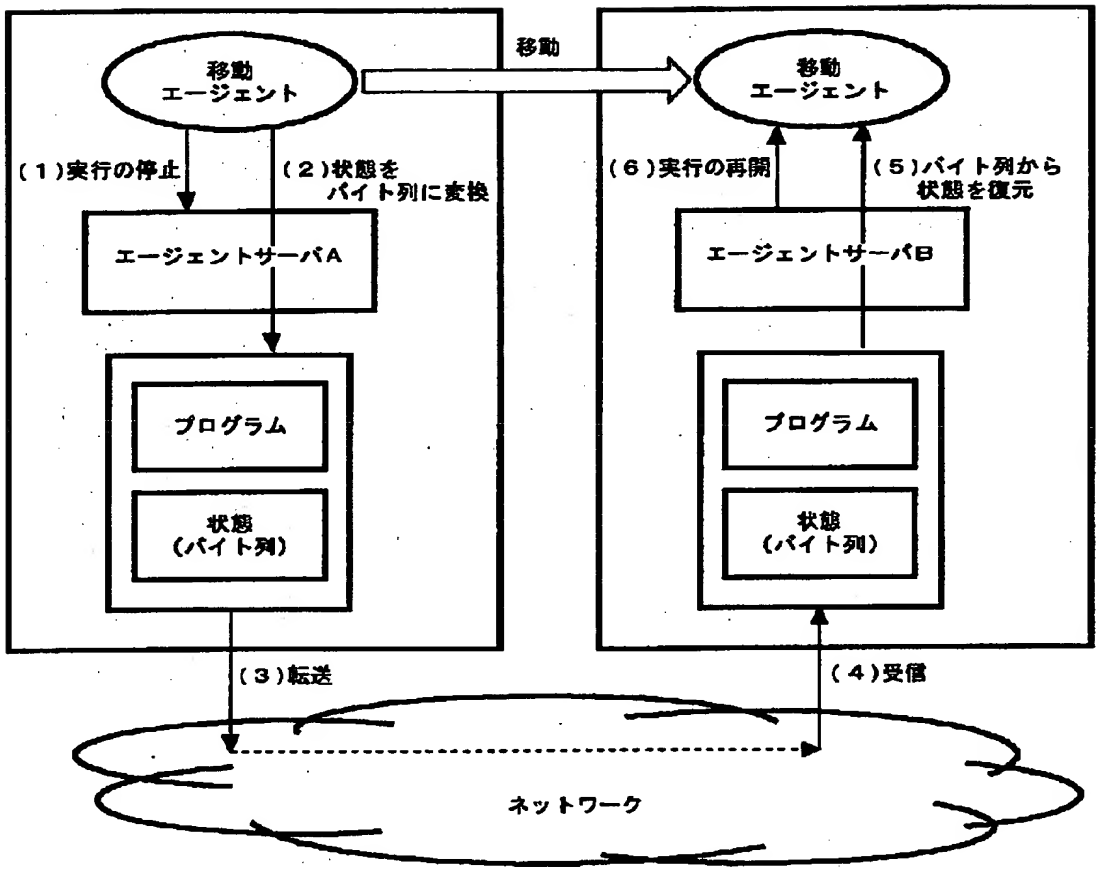
【書類名】　図面

【図１】

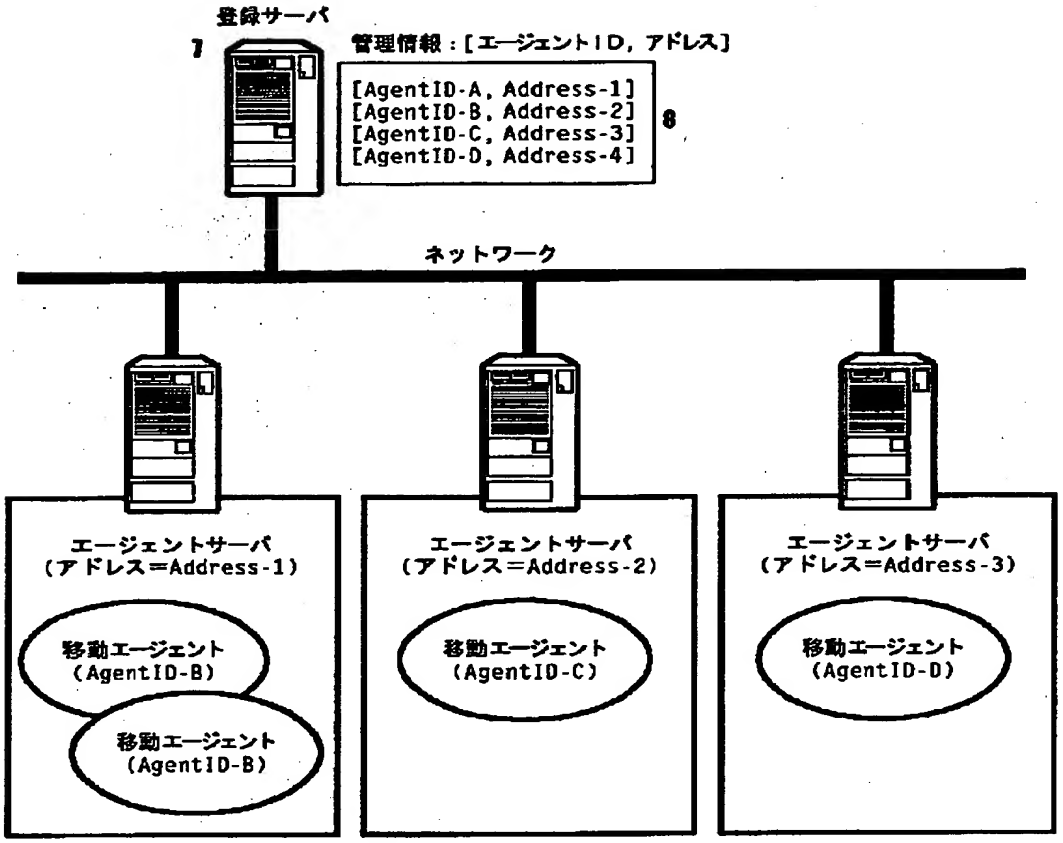


出証特２０００－３０１４４６５

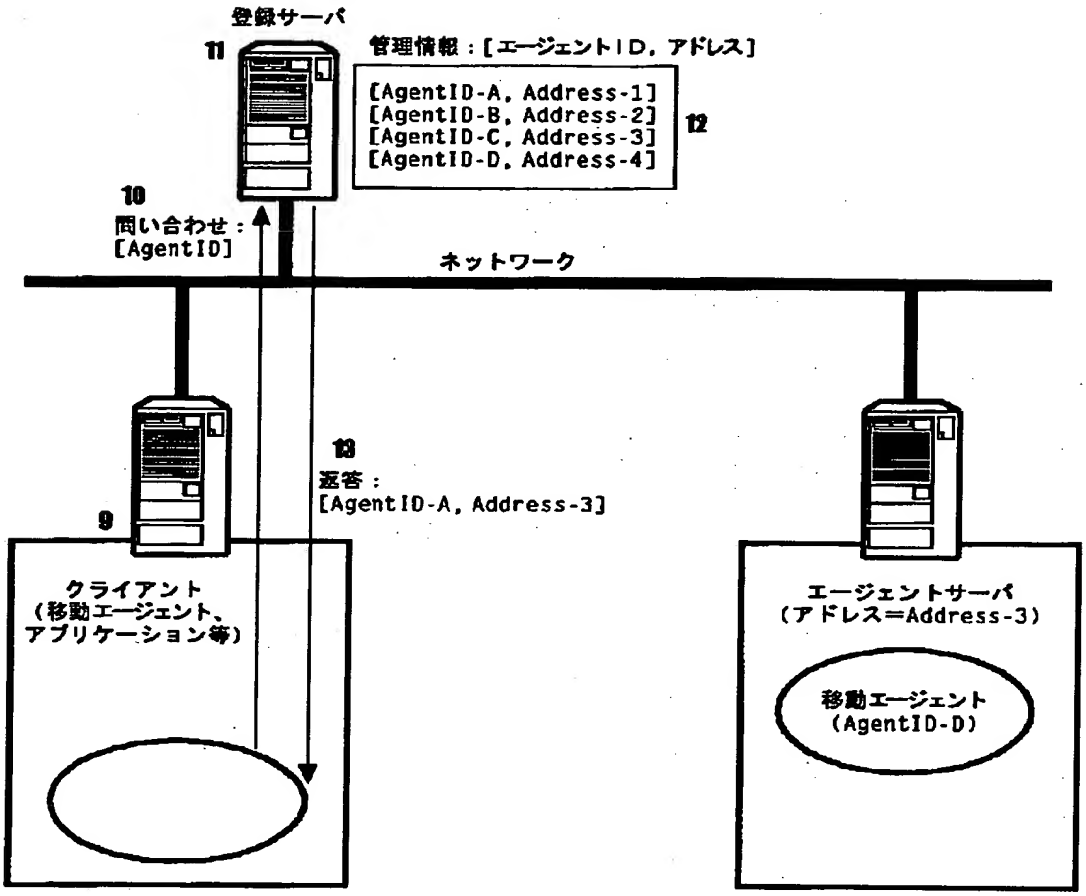
【図２】


移動
移動
エージェント
移動
エージェント
(１)実行の停止
(２)状態を
バイト列に変換
(６)実行の再開
(５)バイト列から
状態を復元
エージェントサーバA
エージェントサーバB
プログラム
状態
(バイト列)
プログラム
状態
(バイト列)
(３)転送
(４)受信
ネットワーク

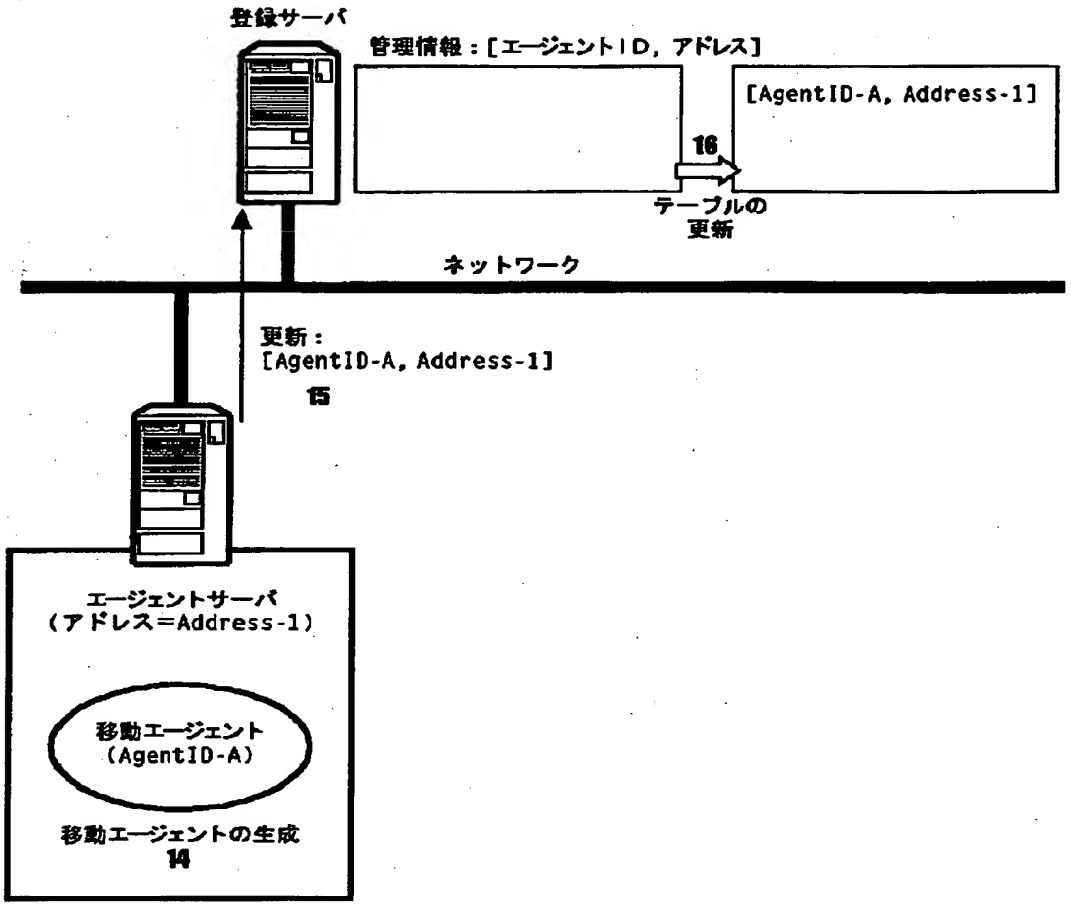
【図３】


登録サーバ
7
管理情報：[エージェントID，アドレス]
[AgentID-A, Address-1]
[AgentID-B, Address-2]
[AgentID-C, Address-3]
[AgentID-D, Address-4]
8
ネットワーク
エージェントサーバ
(アドレス＝Address-1)
移動エージェント
(AgentID-B)
移動エージェント
(AgentID-B)
エージェントサーバ
(アドレス＝Address-2)
移動エージェント
(AgentID-C)
エージェントサーバ
(アドレス＝Address-3)
移動エージェント
(AgentID-D)

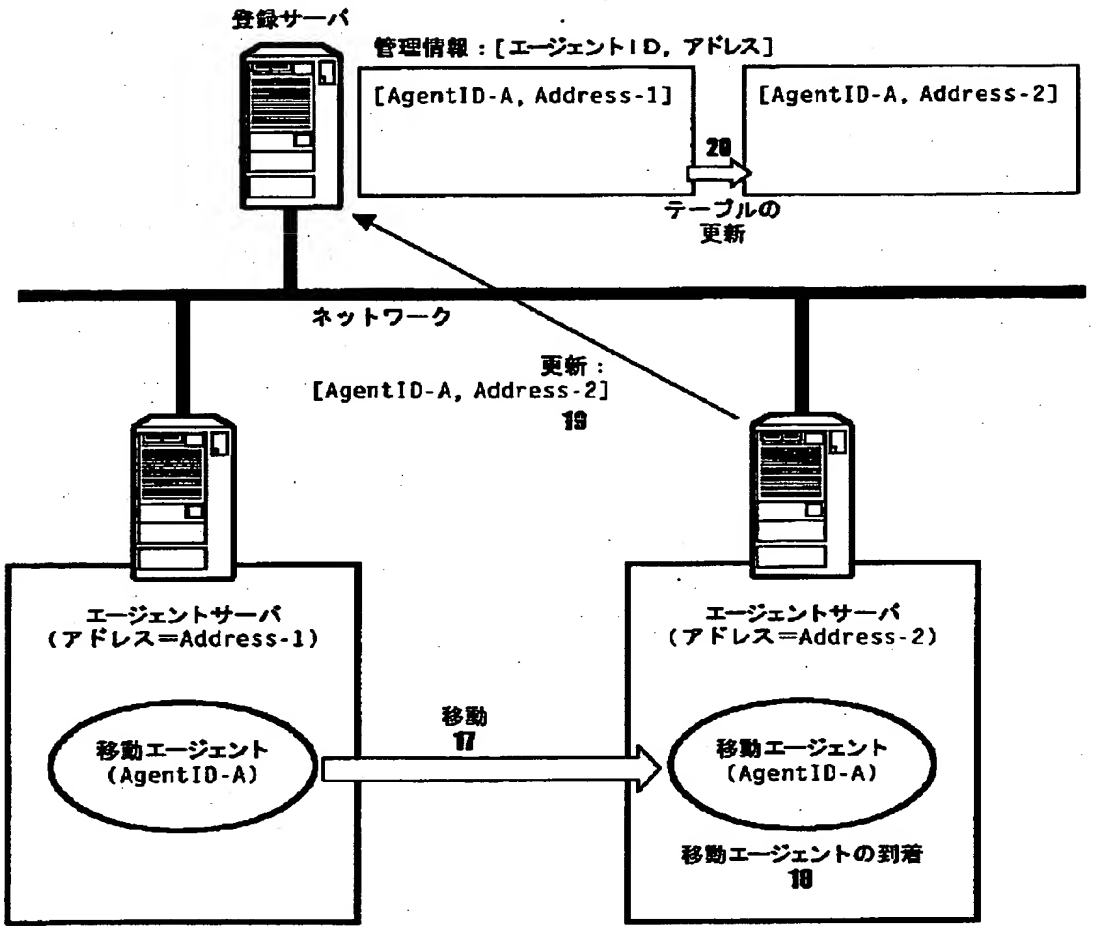
【図４】


登録サーバ
11
管理情報：[エージェントID，アドレス]
[AgentID-A，Address-1]
[AgentID-B，Address-2]
[AgentID-C，Address-3]
[AgentID-D，Address-4]
12
10
問い合わせ：
[AgentID]
ネットワーク
13
返答：
[AgentID-A，Address-3]
9
クライアント
（移動エージェント、
アプリケーション等）
エージェントサーバ
（アドレス＝Address-3）
移動エージェント
（AgentID-D）

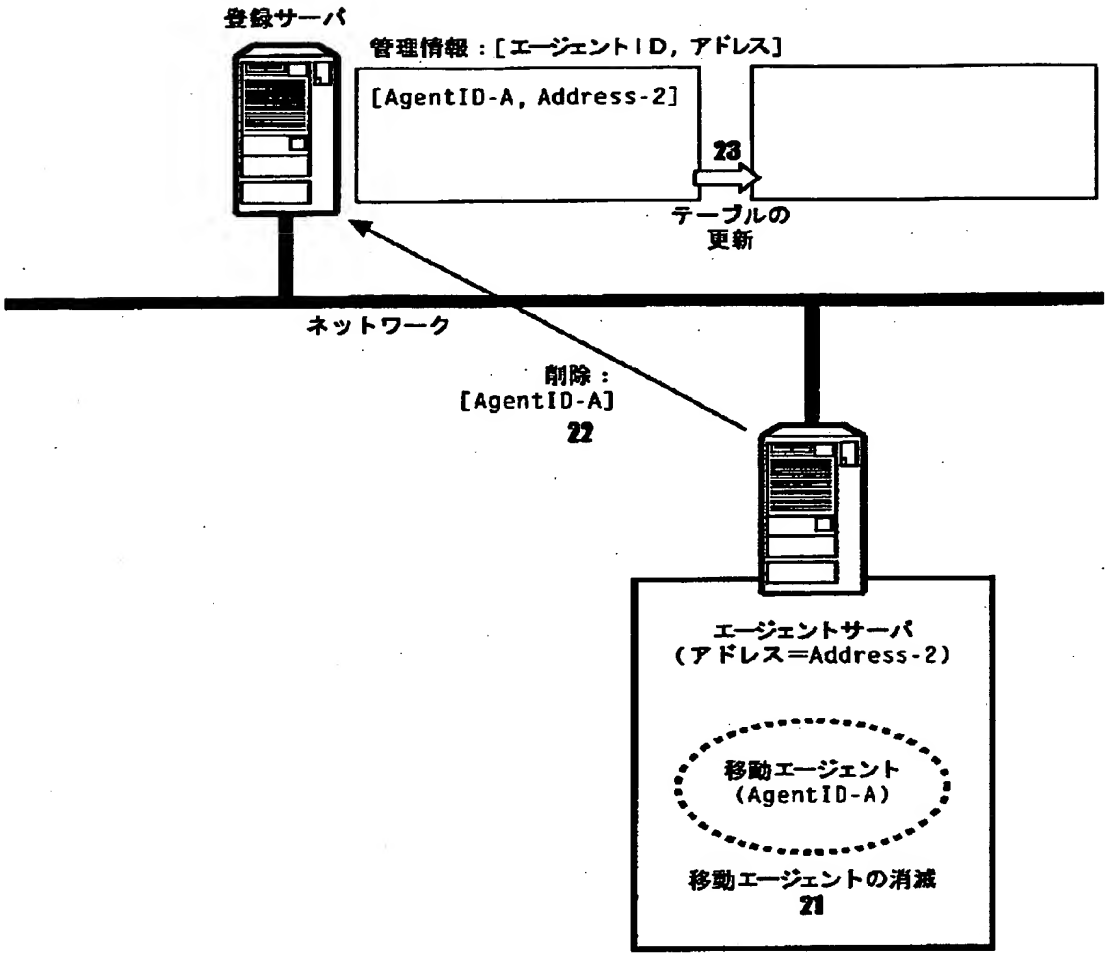
【図５】


登録サーバ
管理情報：[エージェントID，アドレス]
[AgentID-A，Address-1]
16
テーブルの
更新
ネットワーク
更新：
[AgentID-A，Address-1]
15
エージェントサーバ
（アドレス＝Address-1）
移動エージェント
（AgentID-A）
移動エージェントの生成
14

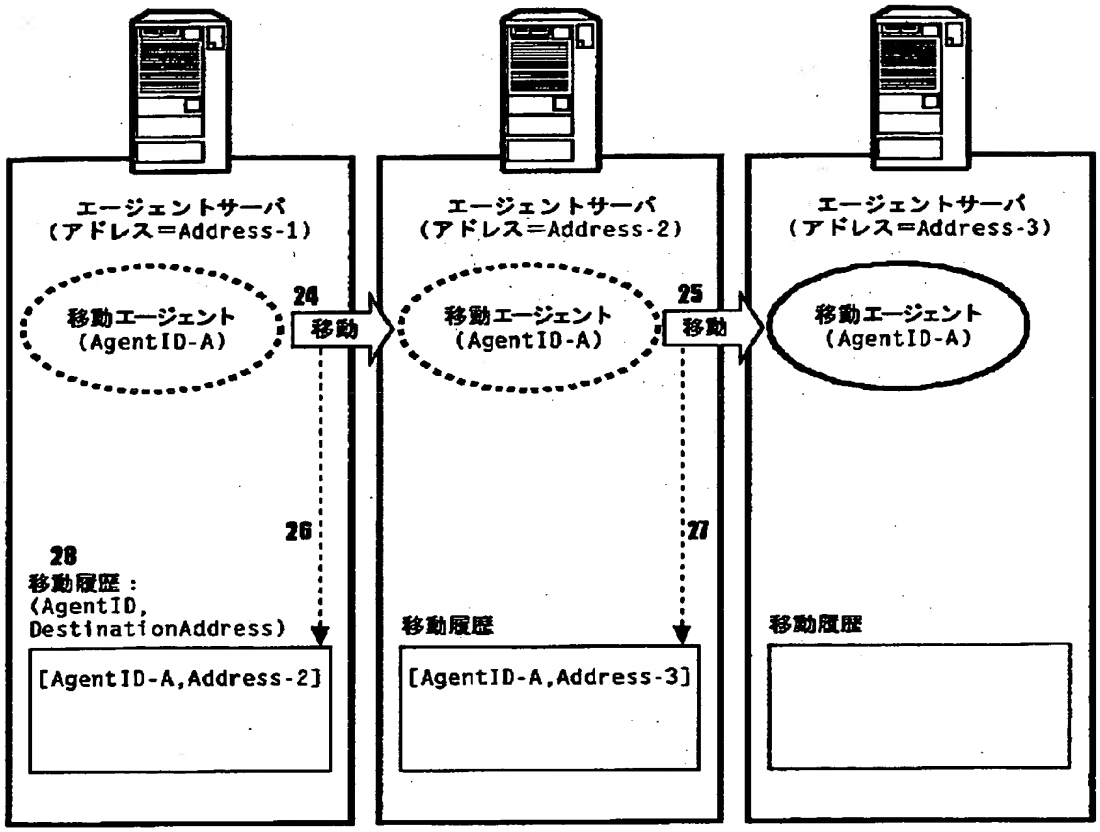
【図６】


登録サーバ
管理情報：[エージェントID，アドレス]
[AgentID-A, Address-1]
[AgentID-A, Address-2]
20
テーブルの
更新
ネットワーク
更新：
[AgentID-A, Address-2]
19
エージェントサーバ
（アドレス＝Address-1）
移動エージェント
(AgentID-A)
移動
17
エージェントサーバ
（アドレス＝Address-2）
移動エージェント
(AgentID-A)
移動エージェントの到着
18

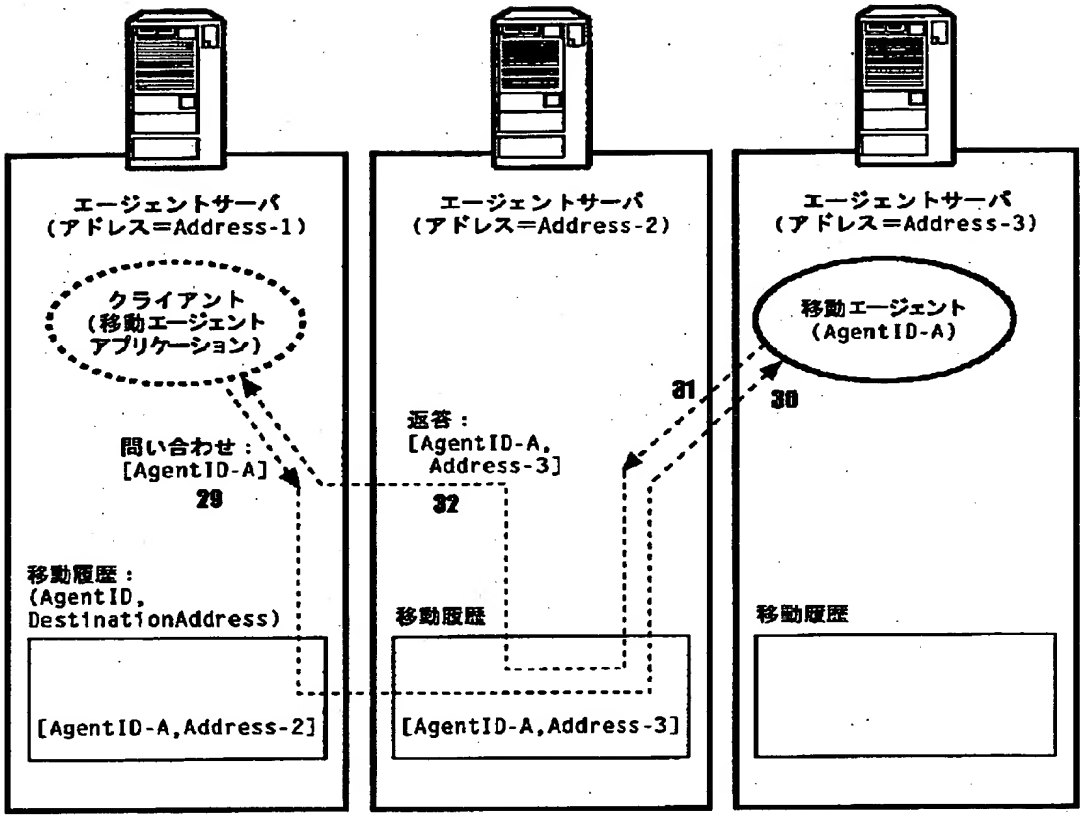
【図７】


登録サーバ
管理情報：[エージェントID，アドレス]
[AgentID-A, Address-2]
23
テーブルの
更新
ネットワーク
削除：
[AgentID-A]
22
エージェントサーバ
（アドレス＝Address-2）
移動エージェント
(AgentID-A)
移動エージェントの消滅
21

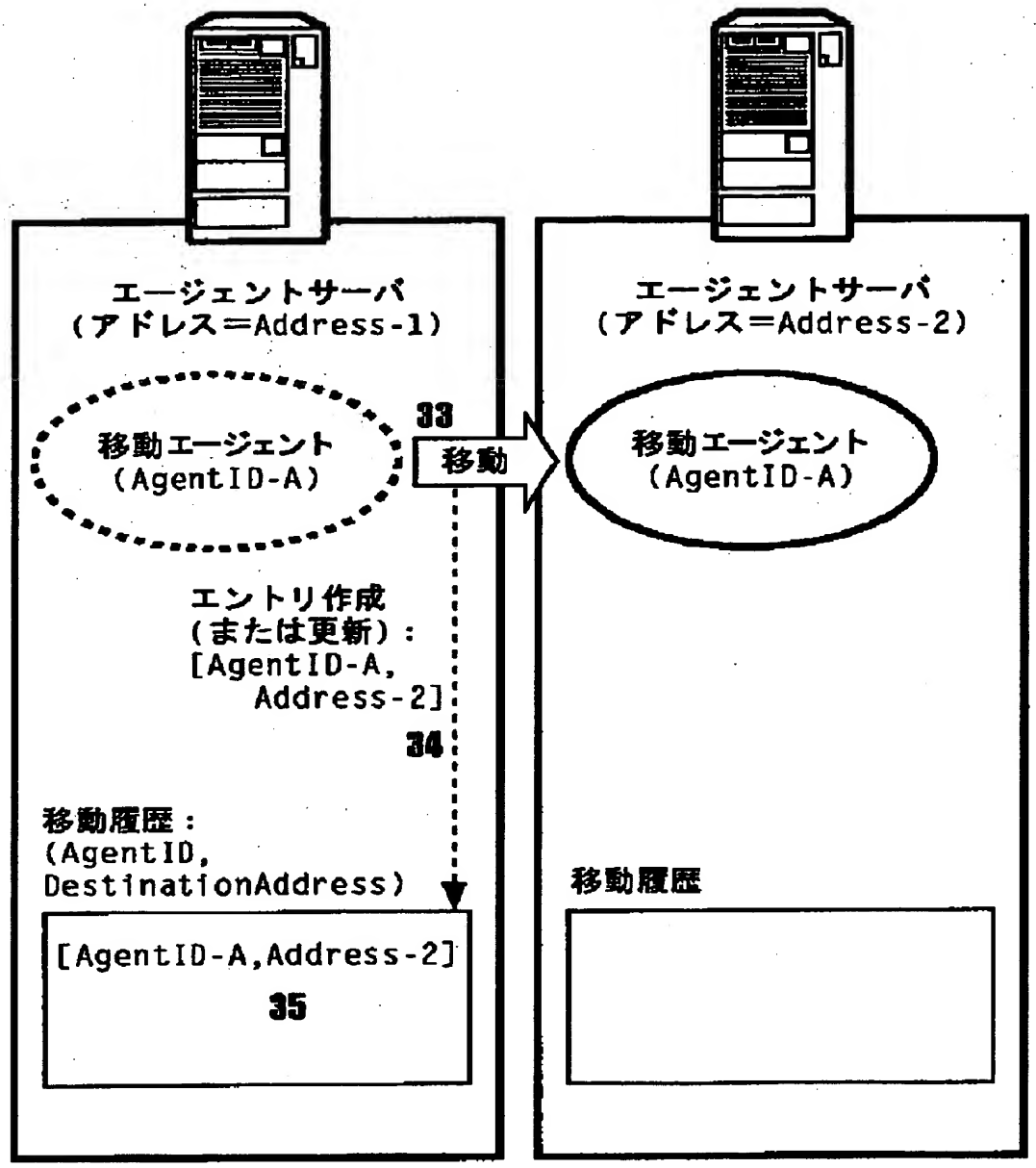
【図８】


エージェントサーバ
(アドレス=Address-1)
移動エージェント
(AgentID-A)
24
移動
エージェントサーバ
(アドレス=Address-2)
移動エージェント
(AgentID-A)
25
移動
エージェントサーバ
(アドレス=Address-3)
移動エージェント
(AgentID-A)
26
27
28
移動履歴：
(AgentID,
DestinationAddress)
[AgentID-A,Address-2]
移動履歴
[AgentID-A,Address-3]
移動履歴

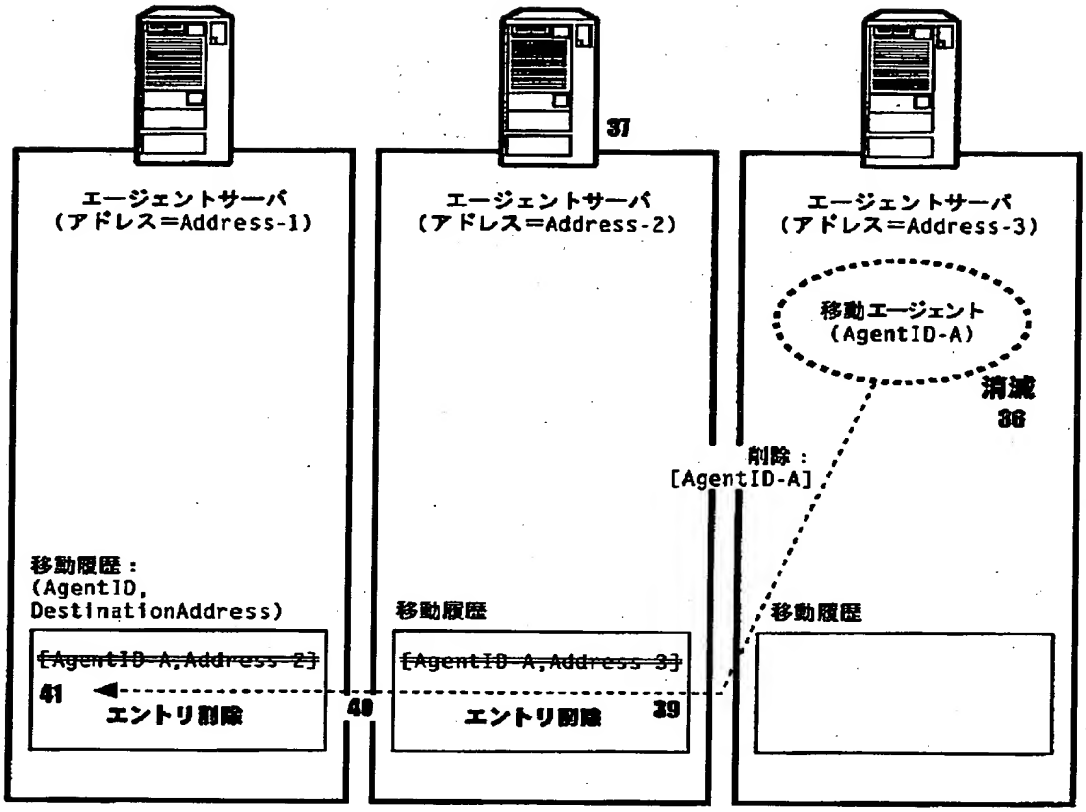
【図９】


エージェントサーバ
(アドレス＝Address-1)
クライアント
(移動エージェント
アプリケーション)
問い合わせ：
[AgentID-A]
29
移動履歴：
(AgentID,
DestinationAddress)
[AgentID-A,Address-2]
エージェントサーバ
(アドレス＝Address-2)
返答：
[AgentID-A,
Address-3]
32
移動履歴
[AgentID-A,Address-3]
エージェントサーバ
(アドレス＝Address-3)
移動エージェント
(AgentID-A)
31
30
移動履歴

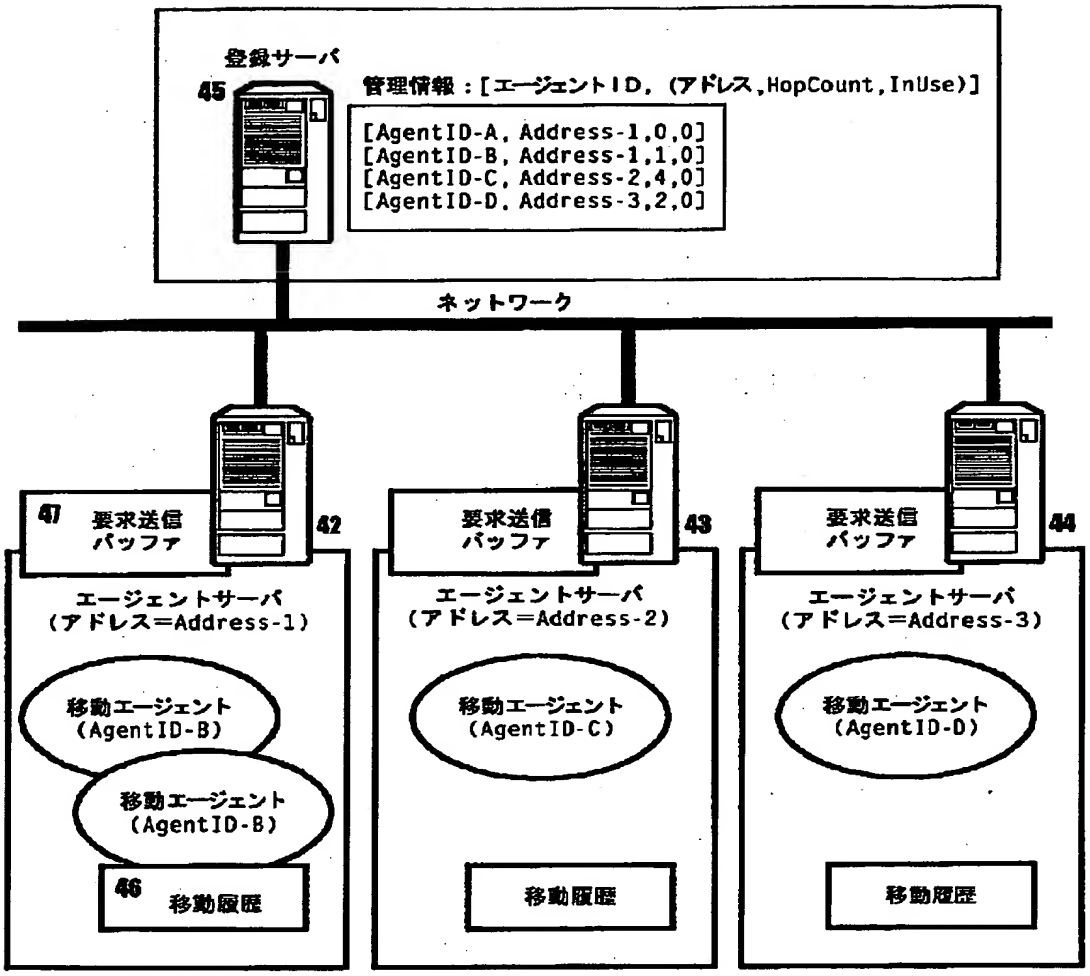
【図１０】


エージェントサーバ
(アドレス＝Address-1)
エージェントサーバ
(アドレス＝Address-2)
移動エージェント
(AgentID-A)
33
移動
移動エージェント
(AgentID-A)
エントリ作成
(または更新)：
[AgentID-A,
Address-2]
34
移動履歴：
(AgentID,
DestinationAddress)
移動履歴
[AgentID-A,Address-2]
35

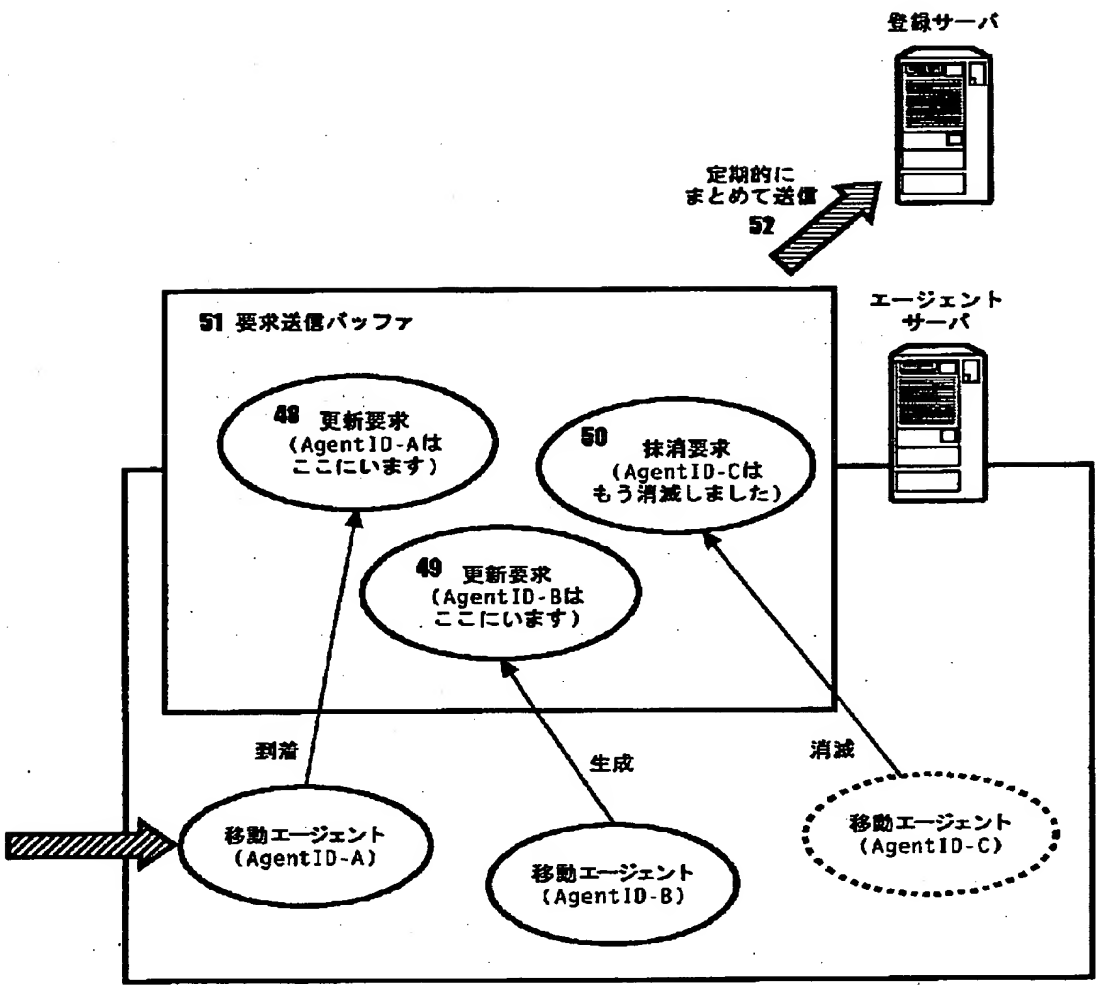
【図１１】


エージェントサーバ
(アドレス=Address-1)
エージェントサーバ
(アドレス=Address-2)
37
エージェントサーバ
(アドレス=Address-3)
移動エージェント
(AgentID-A)
消滅
36
削除：
[AgentID-A]
移動履歴：
(AgentID,
DestinationAddress)
移動履歴
移動履歴
{AgentID-A,Address-2}
{AgentID-A,Address-3}
41
エントリ削除
40
エントリ削除
39

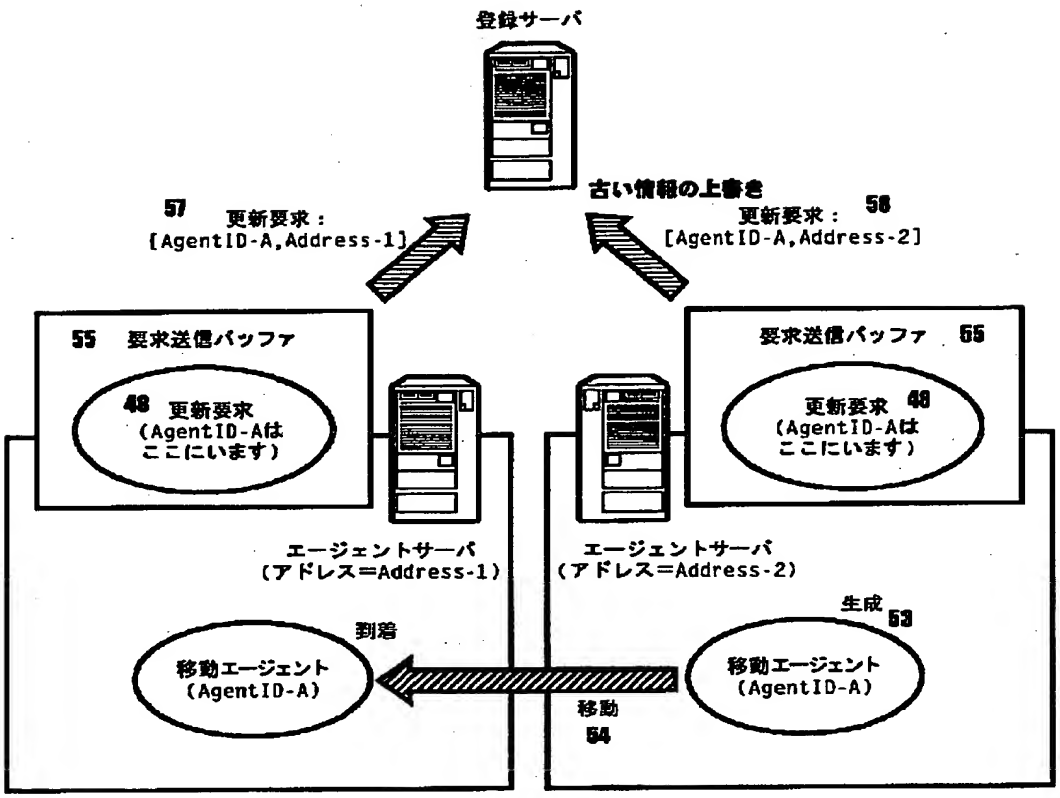
【図１２】


登録サーバ
45
管理情報：[エージェントID，(アドレス,HopCount,InUse)]
[AgentID-A, Address-1,0,0]
[AgentID-B, Address-1,1,0]
[AgentID-C, Address-2,4,0]
[AgentID-D, Address-3,2,0]
ネットワーク
47
要求送信
バッファ
42
エージェントサーバ
(アドレス=Address-1)
移動エージェント
(AgentID-B)
移動エージェント
(AgentID-B)
46
移動履歴
要求送信
バッファ
43
エージェントサーバ
(アドレス=Address-2)
移動エージェント
(AgentID-C)
移動履歴
要求送信
バッファ
44
エージェントサーバ
(アドレス=Address-3)
移動エージェント
(AgentID-D)
移動履歴

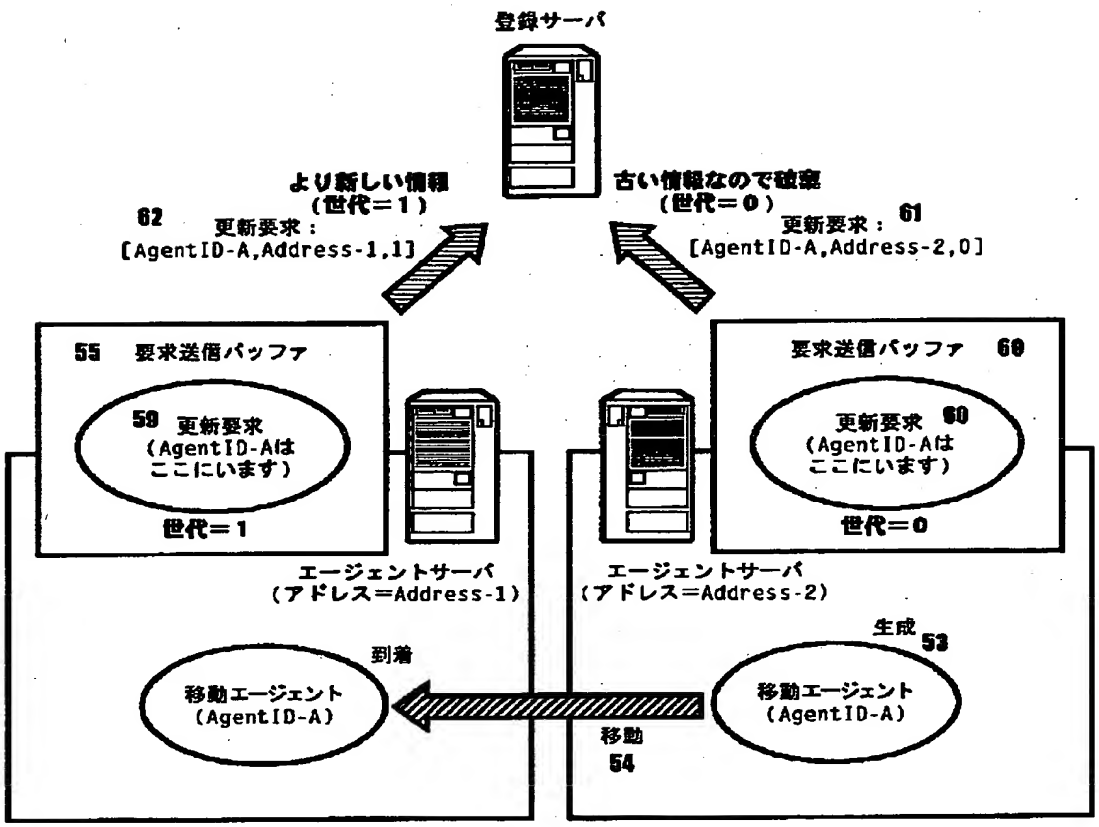
【図１３】


登録サーバ
定期的に
まとめて送信
52
エージェント
サーバ
51 要求送信バッファ
48 更新要求
(AgentID-Aは
ここにいます)
50 抹消要求
(AgentID-Cは
もう消滅しました)
49 更新要求
(AgentID-Bは
ここにいます)
到着
生成
消滅
移動エージェント
(AgentID-A)
移動エージェント
(AgentID-B)
移動エージェント
(AgentID-C)

【図１４】


登録サーバ
古い情報の上書き
57
更新要求：
[AgentID-A,Address-1]
58
更新要求：
[AgentID-A,Address-2]
55 要求送信バッファ
48 更新要求
(AgentID-Aは
ここにいます)
要求送信バッファ 55
更新要求 48
(AgentID-Aは
ここにいます)
エージェントサーバ
(アドレス=Address-1)
エージェントサーバ
(アドレス=Address-2)
到着
移動エージェント
(AgentID-A)
生成
53
移動エージェント
(AgentID-A)
移動
54

【図１５】



【図１６】

要求バッファに格納する要求

```
[AgentID, (Kind, HopCount)]

Agent
    対象とする移動エージェントのＩＤ
Kind
    要求の種類
        REGIST：生成、到着時（到着サーバの管理テーブルの更新を要求）
        UNREGIST：消滅時（登録サーバの管理テーブルからエントリの削除を要求）
HopCount
    要求の世代
        要求を生成したときに、移動エージェントの積算移動回数
```

【図１７】

登録サーバの管理テーブルのエントリ

[AgentID, (Address, HopCount, InUse)]

AgentID
　対象とする移動エージェントのＩＤ
Address
　移動エージェントの追跡開始地点となるエージェントサーバのアドレス
HopCount
　要求の世代
　　要求を生成したときに、移動エージェントの積算移動回数
InUse
　エントリの使用カウント
　　１以上のときは使用中を示す
　　追跡中にエントリや移動履歴が削除されるのを防ぐために使用

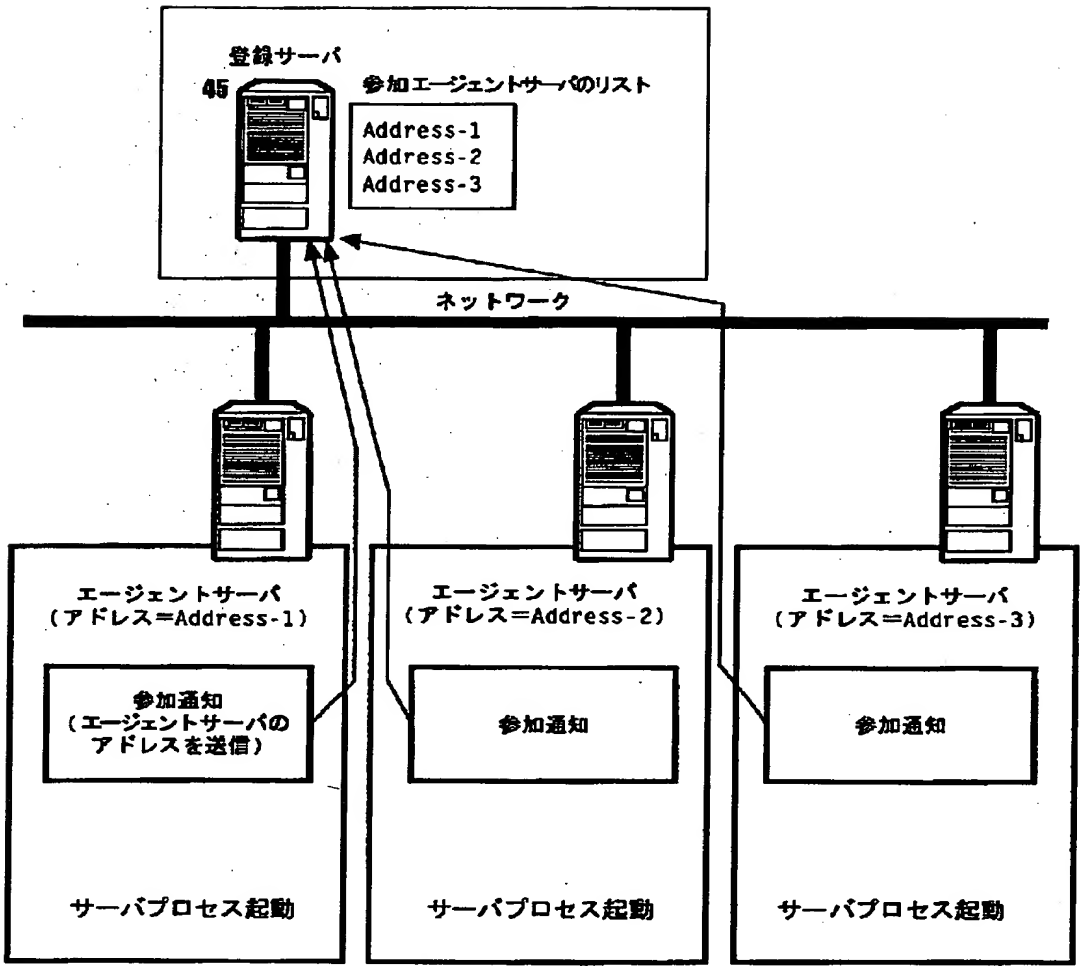
【図１８】

移動履歴

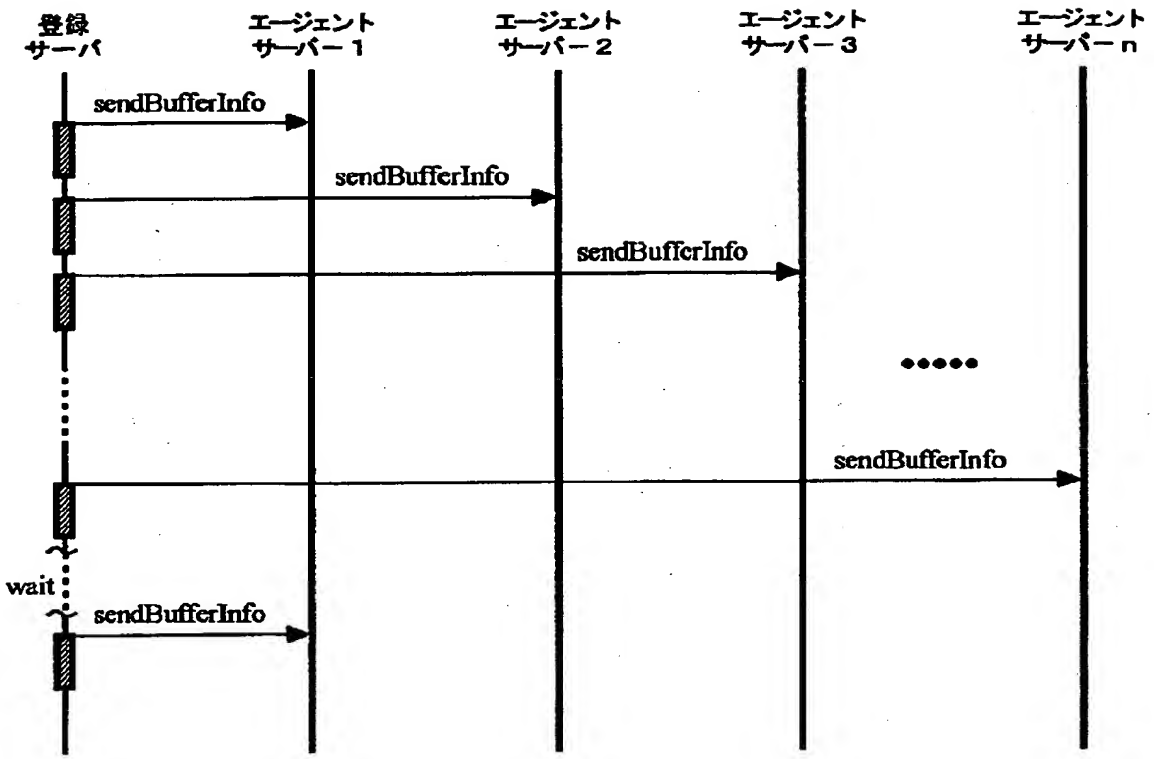
[AgentID, Destination]

AgentID
　対象とする移動エージェントのＩＤ
Address
　移動エージェントの異動先エージェントサーバのアドレス

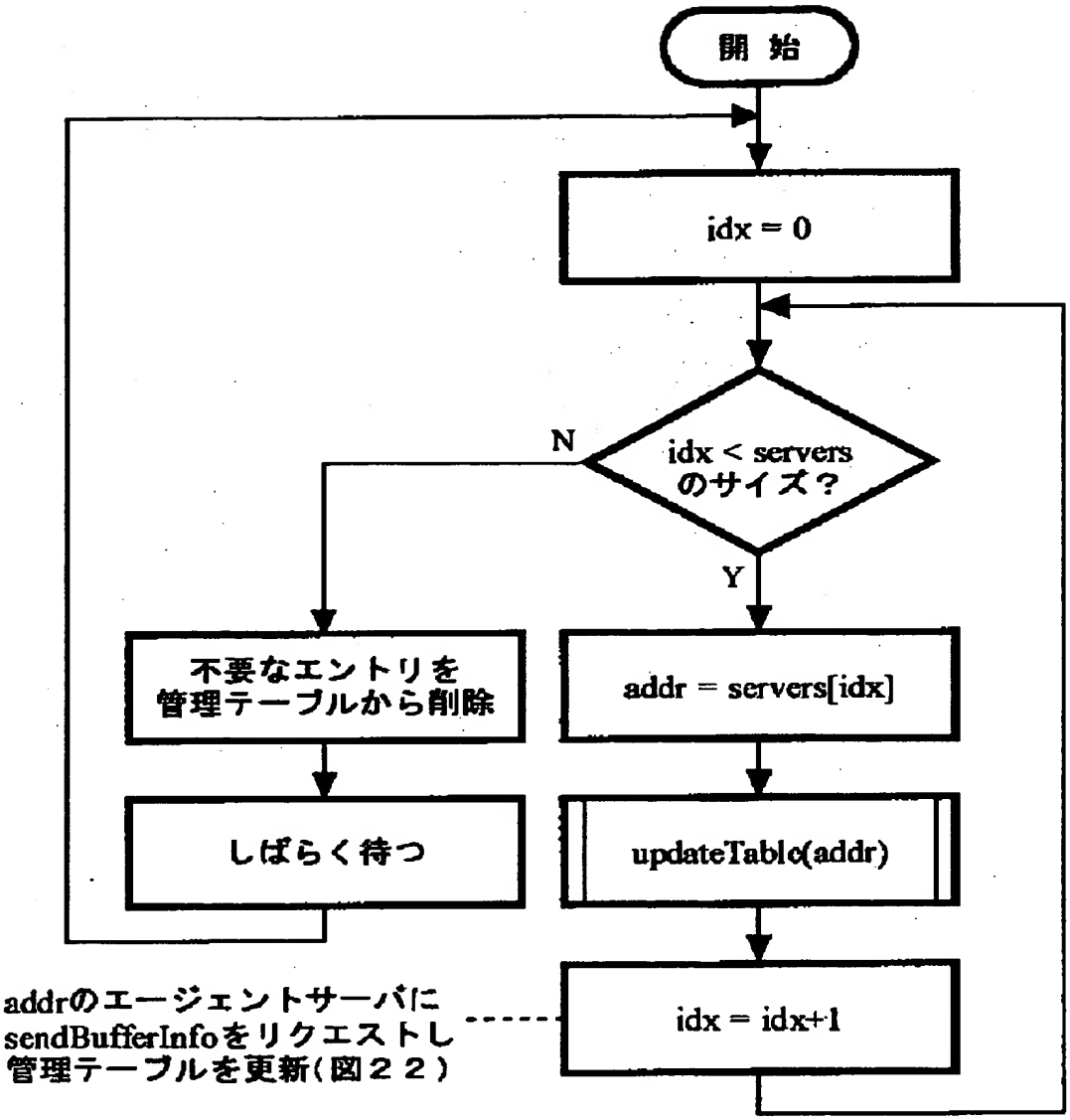
【図１９】


登録サーバ
45
参加エージェントサーバのリスト
Address-1
Address-2
Address-3
ネットワーク
エージェントサーバ
（アドレス＝Address-1）
参加通知
（エージェントサーバの
アドレスを送信）
サーバプロセス起動
エージェントサーバ
（アドレス＝Address-2）
参加通知
サーバプロセス起動
エージェントサーバ
（アドレス＝Address-3）
参加通知
サーバプロセス起動

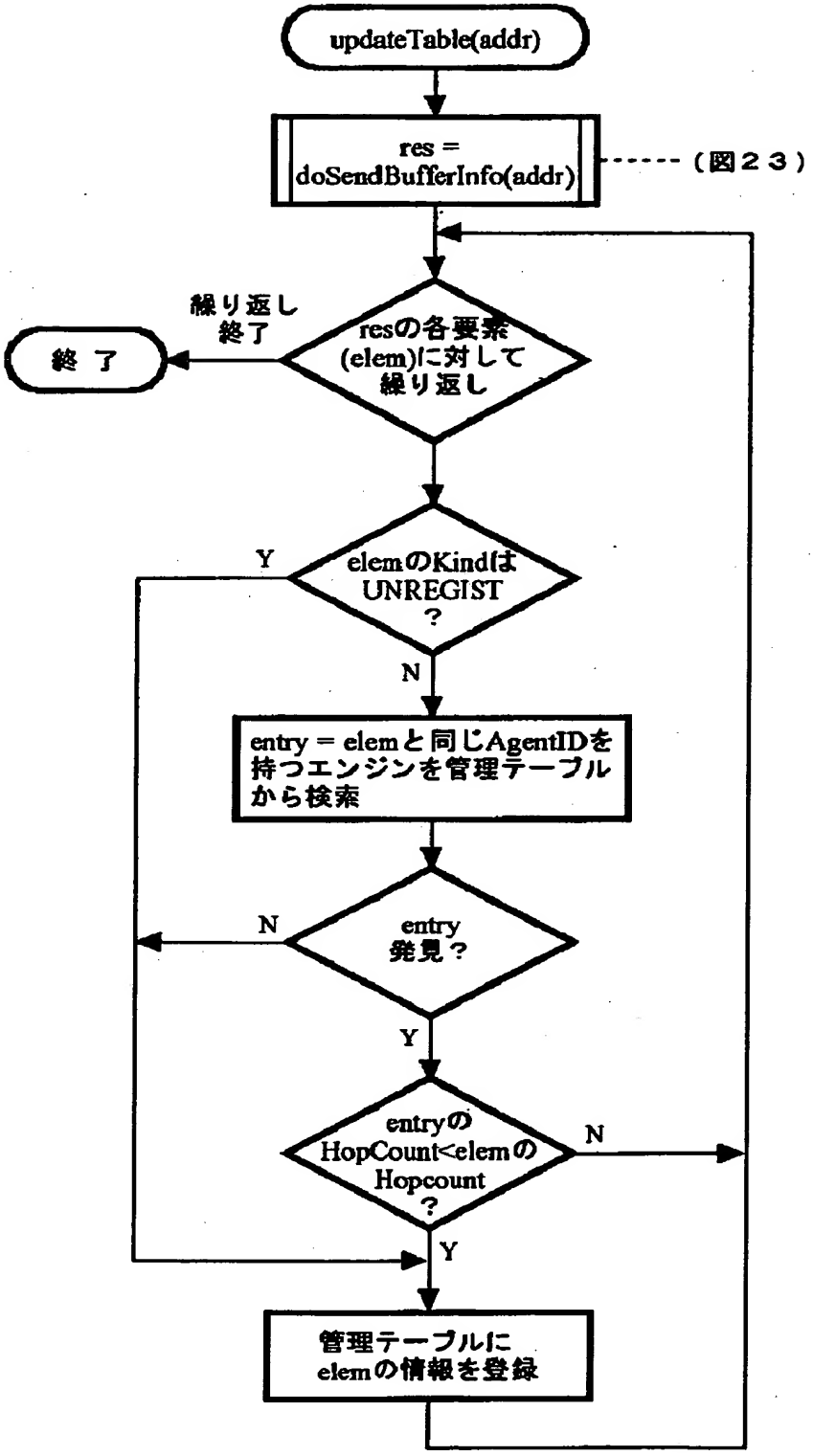
【図２０】


登録
サーバ
エージェント
サーバー１
エージェント
サーバー２
エージェント
サーバー３
エージェント
サーバーｎ
sendBufferInfo
sendBufferInfo
sendBufferInfo
sendBufferInfo
wait
sendBufferInfo

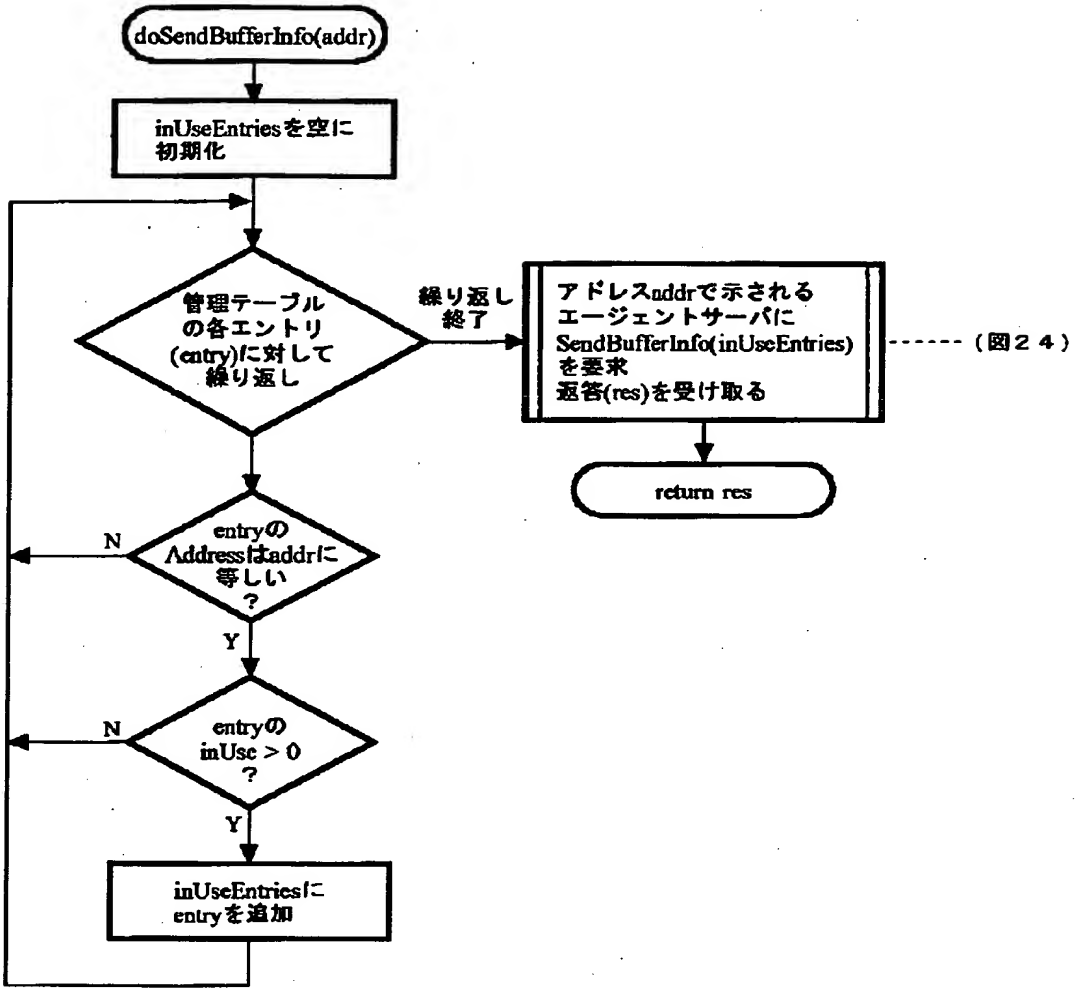
【図２１】


開 始
idx = 0
idx < servers のサイズ？
N
Y
不要なエントリを 管理テーブルから削除
addr = servers[idx]
しばらく待つ
updateTable(addr)
idx = idx+1
addrのエージェントサーバに sendBufferInfoをリクエストし 管理テーブルを更新(図２２)

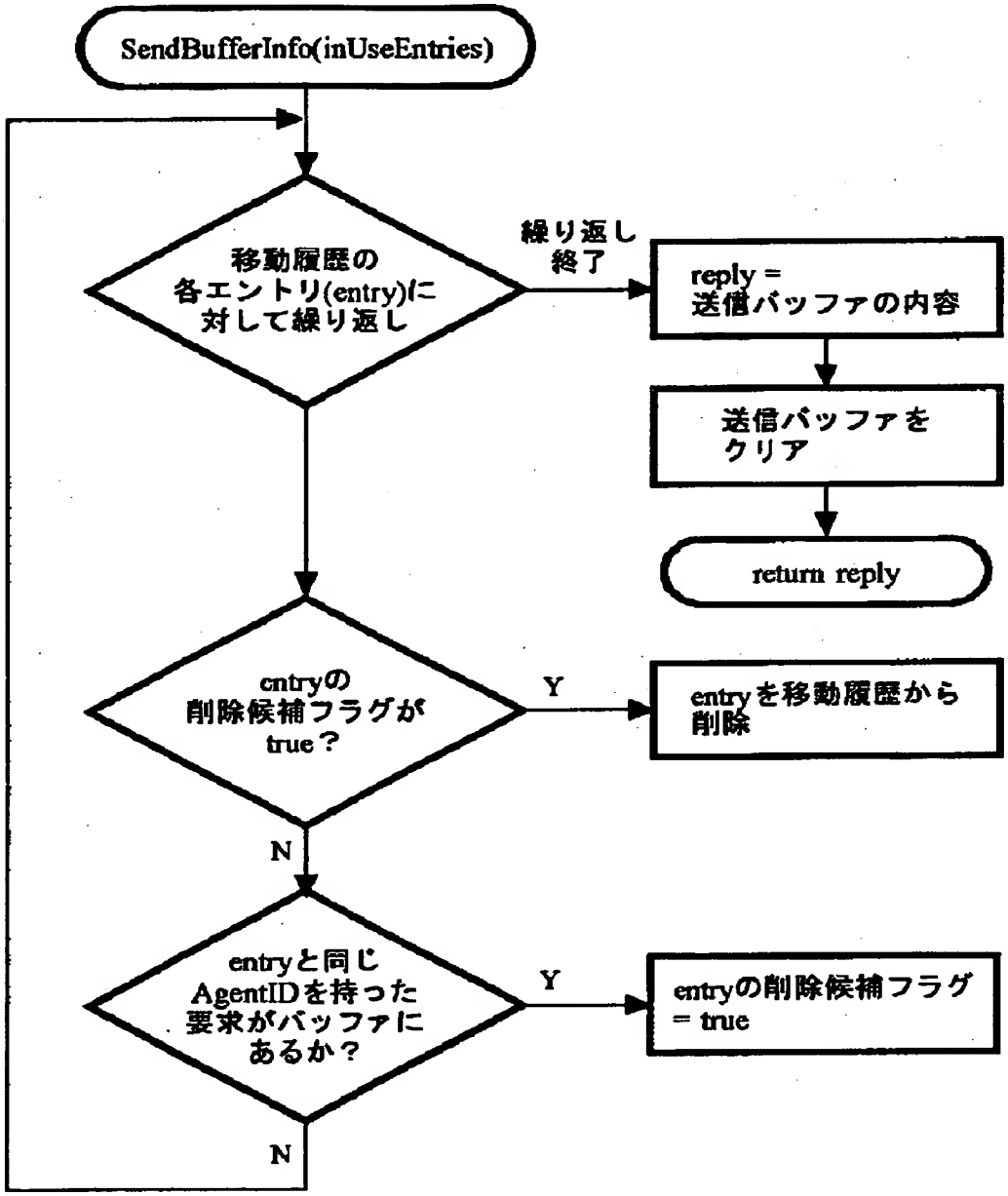
【図２２】


updateTable(addr)
res = doSendBufferInfo(addr)
（図２３）
繰り返し終了
終了
resの各要素(elem)に対して繰り返し
Y
elemのKindはUNREGIST？
N
entry = elemと同じAgentIDを持つエンジンを管理テーブルから検索
N
entry発見？
Y
entryのHopCount<elemのHopcount？
N
Y
管理テーブルにelemの情報を登録

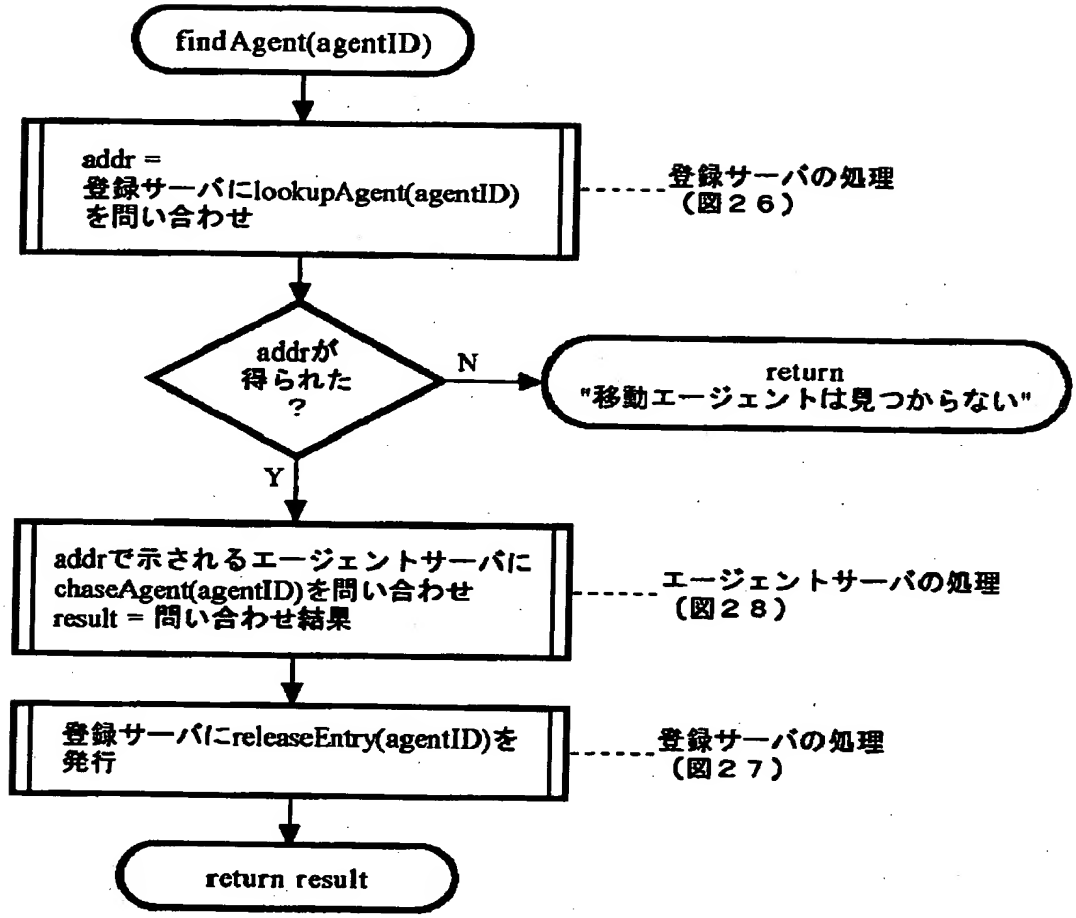
【図２３】


doSendBufferInfo(addr)
inUseEntriesを空に
初期化
管理テーブル
の各エントリ
(entry)に対して
繰り返し
繰り返し
終了
アドレスaddrで示される
エージェントサーバに
SendBufferInfo(inUseEntries)
を要求
返答(res)を受け取る
（図２４）
return res
entryの
Addressはaddrに
等しい
？
N
Y
entryの
inUse > 0
？
N
Y
inUseEntriesに
entryを追加

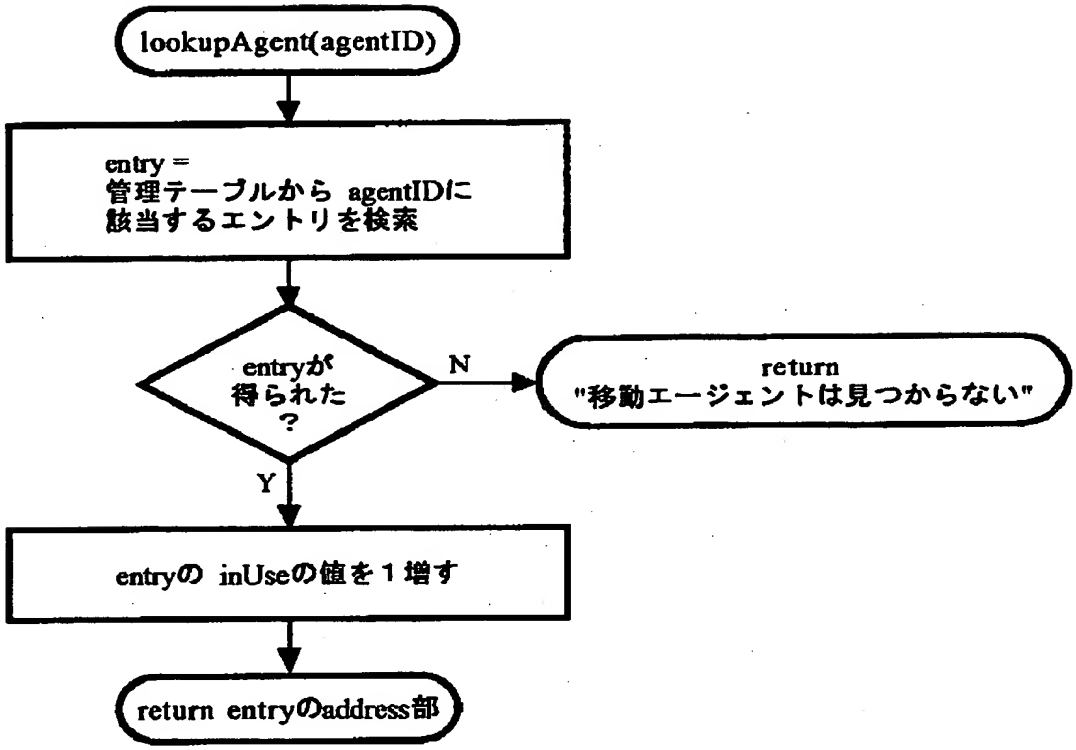
【図２４】


SendBufferInfo(inUseEntries)
移動履歴の
各エントリ(entry)に
対して繰り返し
繰り返し
終了
reply =
送信バッファの内容
送信バッファを
クリア
return reply
entryの
削除候補フラグが
true？
Y
entryを移動履歴から
削除
N
entryと同じ
AgentIDを持った
要求がバッファに
あるか？
Y
entryの削除候補フラグ
= true
N

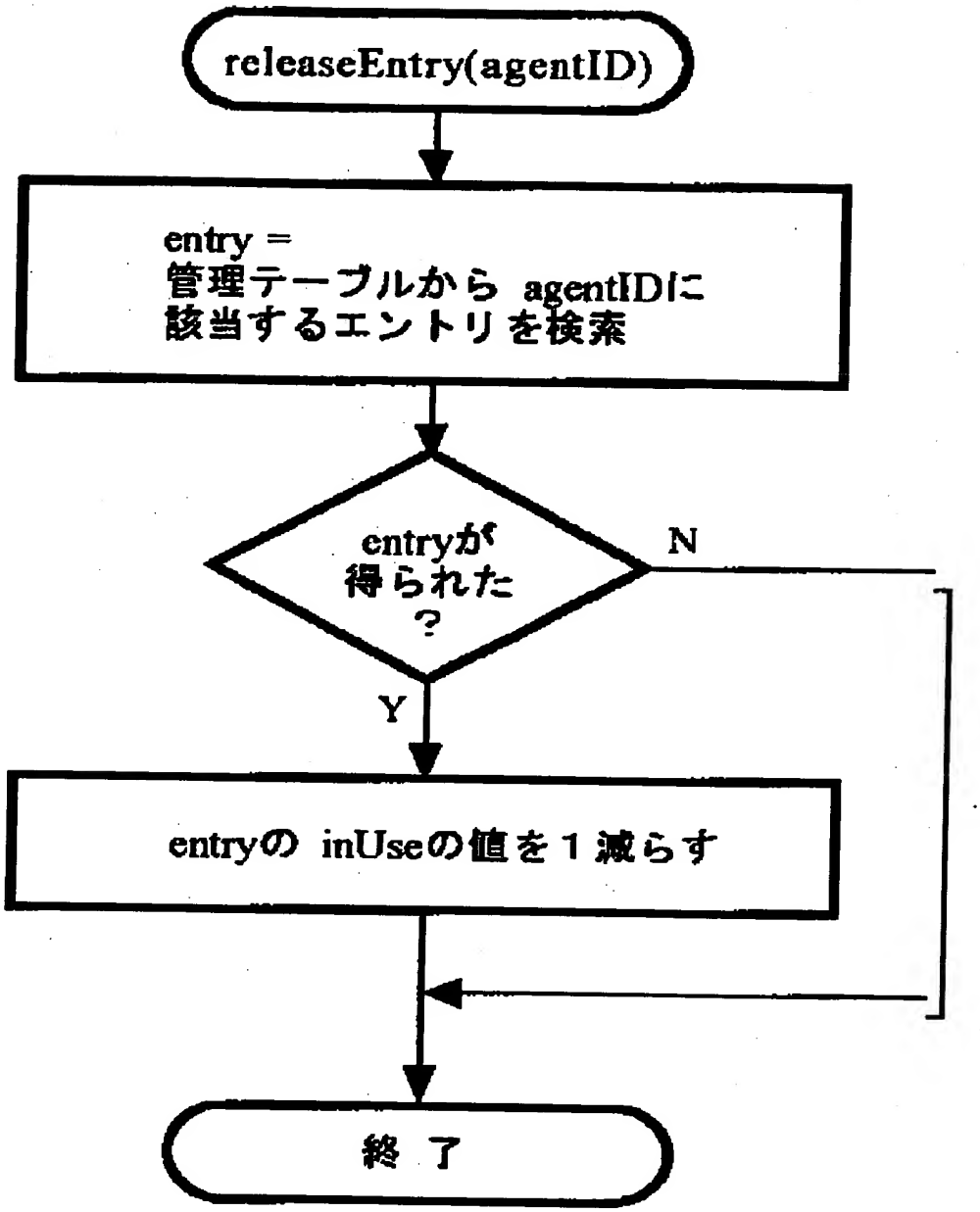
【図２５】


findAgent(agentID)
addr =
登録サーバにlookupAgent(agentID)
を問い合わせ
登録サーバの処理
（図２６）
addrが
得られた
？
N
return
"移動エージェントは見つからない"
Y
addrで示されるエージェントサーバに
chaseAgent(agentID)を問い合わせ
result = 問い合わせ結果
エージェントサーバの処理
（図２８）
登録サーバにreleaseEntry(agentID)を
発行
登録サーバの処理
（図２７）
return result

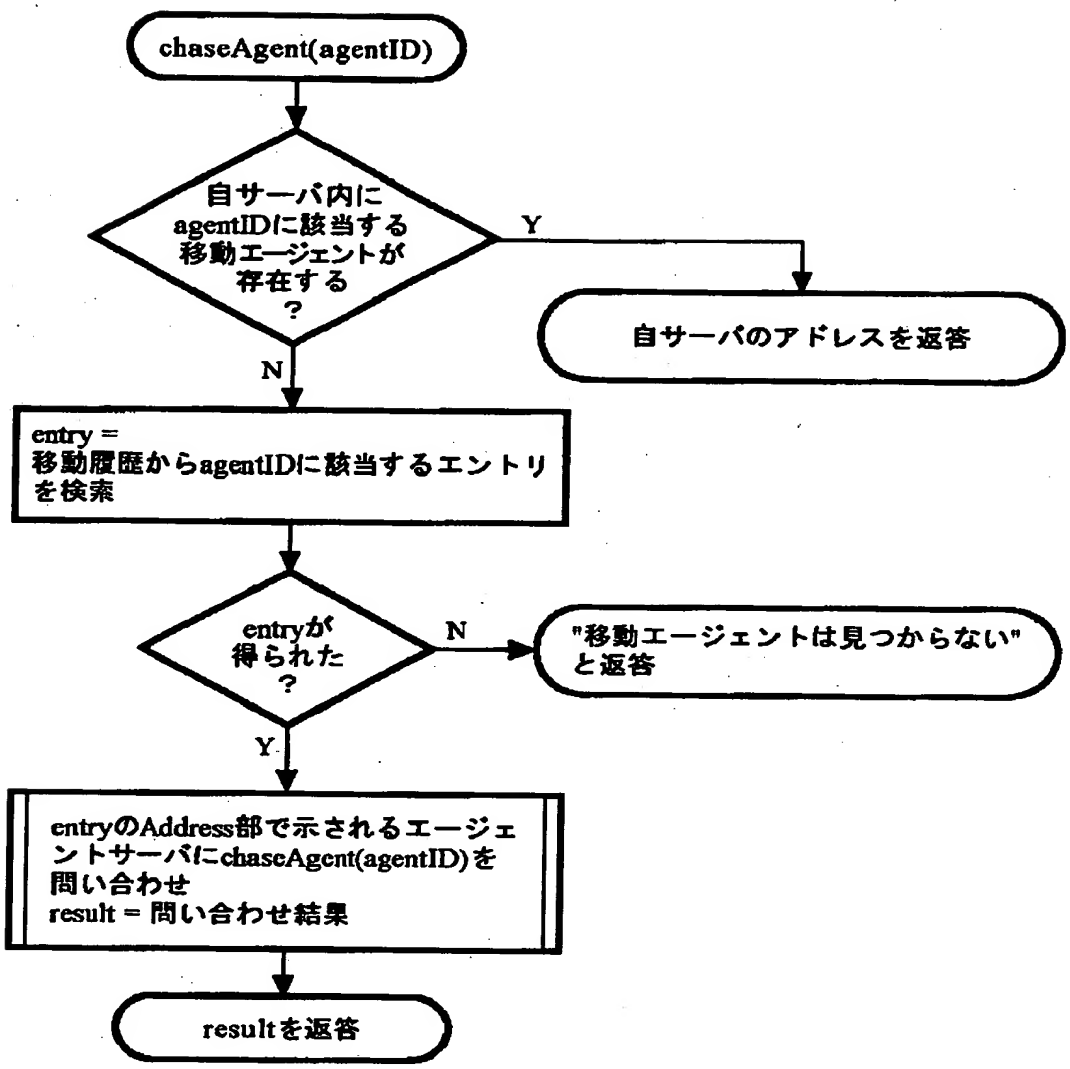
【図２６】


lookupAgent(agentID)
entry =
管理テーブルから agentIDに
該当するエントリを検索
entryが
得られた
？
N
return
"移動エージェントは見つからない"
Y
entryの inUseの値を１増す
return entryのaddress部

【図２７】


releaseEntry(agentID)
entry =
管理テーブルから agentIDに
該当するエントリを検索
entryが
得られた
？
N
Y
entryの inUseの値を１減らす
終 了

【図２８】


chaseAgent(agentID)
自サーバ内にagentIDに該当する移動エージェントが存在する？
Y
自サーバのアドレスを返答
N
entry = 移動履歴からagentIDに該当するエントリを検索
entryが得られた？
N
"移動エージェントは見つからない"と返答
Y
entryのAddress部で示されるエージェントサーバにchaseAgent(agentID)を問い合わせ
result = 問い合わせ結果
resultを返答

【図２９】

ケース１：
Ｎ＝９（エージェント総数９０）

従来の方法 (Registry)	本方法	追跡機構なし
79,483	63,599	61,084
80,008	66,079	65,241
90,480	66,079	66,504

【図３０】

ケース２：
Ｎ＝１５（エージェント総数１５０）

従来の方法 (Registry)	本方法	追跡機構なし
145,356	108,240	97,802
153,234	107,596	108,289
179,300	109,654	112,966

【書類名】　要約書

【要約】

【課題】より高い次元で移動エージェントの位置管理を行う。

【解決手段】　各エージェントサーバ(42,43,44)は、移動履歴(46)を保持し管理する。各エージェントサーバは、移動エージェントの位置情報の更新要求を、移動エージェントの移動回数と対応づけて一時的に貯え、定期的に登録サーバ(45)に送信するようにして、登録サーバへのアクセスに輻輳が生じないようにする。登録サーバは、より移動回数が多い更新要求のみを、移動サーバの位置情報に反映させ、古い情報でより新しい情報を上書きしないようにする。

【選択図】図１２

認定・付加情報

特許出願の番号	平成１１年　特許願　第３２７２７６号
受付番号	５９９０１１２５６９１
書類名	特許願
担当官	第七担当上席　　００９６
作成日	平成１１年１１月１９日

＜認定情報・付加情報＞

【提出日】　平成11年11月17日

次頁無

出証特２０００－３０１４４６５

出　願　人　履　歴　情　報

識別番号　　〔３９０００９５３１〕

１．変更年月日　　１９９０年１０月２４日
［変更理由］　新規登録
住　所　　アメリカ合衆国１０５０４、ニューヨーク州　アーモンク　（番地なし）
氏　名　　インターナショナル・ビジネス・マシーンズ・コーポレイション